Open Communication Interface for Road Traffic Control Systems

Offene Schnittstellen für die Straßenverkehrstechnik

OCIT-Outstations

Basisfunktionen für Feldgeräte

OCIT-O-Basis_V1.1_A02

OCIT Developer Group (ODG)

OCIT-Outstations

Basisfunktionen für Feldgeräte

Dokument: OCIT-O-Basis_V1.1_A02

Herausgeber: OCIT Developer Group (ODG)

Kontakt: www.ocit.org

Inhaltsverzeichnis

Dokumentenstand

Version Zustand	Verteilerkreis	Datum	Kommentar
V 1.0	PUBLIC	6. September 2002	1. freigegebene Ausgabe
V1.1 A01	PUBLIC	15. Juli 2004	Text an verschiedenen Stellen aktualisiert und korrigiert. Kapitel überarbeitet: 2.1 Systemzeit, 2.2.1.2 Methode OnNetzAus (schnelle Meldung bei Netzausfall), 3 Systemzugänge, 4.1.1 Systemobjekt Feldgerät, 4.2.4.3 Eventhandler für Listen in der Zentrale, 4.2.6.2 Meldungsauftrag, 4.2.12 Erweiterung der Türmeldungen, 4.2.4.3 Eventhandler (Methode OnNetzAus) Kapitel neu: 2.4 Zählweise bei nummerierten Elementen
V1.1 A02	PUBLIC	17. 11. 2004	Event **OnNetzAus()** ergänzt mit Vorgangskennung (Kapitel 2.2.1.2 und 4.2.4.3, Methode 203).

Referenzdokumente OCIT-Outstations, Stand 17. 11. 2004

Gültig	Dokumente	Titel	Datenspezifikationen (XML-Dateien)
Generell	OCIT-O-System_V1.1	Einführung in das System	
	OCIT-O-Protokoll_V1.1	Regeln und Protokolle	OCIT-O-DTD_V1.0.dtd
	OCIT-O-Basis_V1.1_A02	Basisdefinitionen für Feldgeräte [1]	OCIT-O-Basis-TYPE_V1.1_A02.xml
Speziell	OCIT-O-Lstg_V1.1_A02	Lichtsignalsteuergeräte	OCIT-O-Lstg-TYPE_V1.1.xml
Optional	OCIT-O- Profil_1_V1.1	Profil 1 – Übertragungsprofil für Punkt-zu-Punkt-Verbindungen auf festgeschalteten Übertragungswegen	

[1] Geräte, deren Einsatzort die Straße ist, wie Lichtsignalsteuerungen, Verkehrsmessstellen oder Anzeigesteuerungen, werden in der OCIT-Standardisierung generalisierend als Feldgeräte bezeichnet.

Abkürzungen

		Verwendet / Standard in
bps	bits per second (= bit/s)	
BTPPL	Basis Transport Paket Protokoll Layer	OCIT-Outstations
HDLC	High level Data Link Protocol	ISO
IP	Internet Protocol	RFC 791 (Internet)
OCIT	Open Communication Interface for Road Traffic Control Systems	
OSI	Open Systems Interconnection	ITU-T
PPP	Point to Point Protocol	Internet
RFC	Request for Comment (= Arbeitspapiere, Protokoll-Spezifikationen oder Kommentare zu Netzwerk-Themen)	Internet
SHA-1	Secure Hash Algorithm	Internet
TCP	Transmission Control Protocol	RFC 793 (Internet
UDP	User Datagram Protocol - low end transport service	Internet
V.xx	Standards der ITU-T (International Telecommunications Union), früher CCITT	International Telecommunications Union
XML	eXtensible Markup Language. Herstellerunabhängige Auszeichnungssprache, mit der u.a. eine Schnittstellenbeschreibung verteilter Applikationen realisiert werden kann (spezifiziert durch W3C)	Internet

1 Einführung

In diesem Dokument finden sich Definitionen zu Funktionen, die in Lichtsignalsteuerungen aber auch in Verkehrsmessstellen und anderen typischen Feldgeräten der Straßenverkehrstechnik in ähnlicher Art und Weise vorhanden sind, wie zum Beispiel Archive oder die Meldungen „Tür auf“, „Störung“ etc.

Es ist nicht verpflichtend, dass OCIT-Outstations konforme Geräte alle hier festgelegten Funktionen unterstützen. Sie unterstützen nur diejenigen Funktionen, die für den jeweiligen Zweck und Ausbau notwendig sind.

Die Definitionen gelten für Feldgeräte und Zentralen.

2 Spezielle Definitionen

2.1 Systemzeit

Die Lichtsignalsteuergeräte besitzen lokale Uhren. Ihre genaue Einstellung ist Sache der Feldgeräte, die dazu den Zeitdienst **NTP (RFC 1305)** der Zentrale nutzen können. Der Zeitdienst gleicht durch die Übertragungszeit zwischen Zentrale und Feldgerät bedingte Zeitfehler aus. Weitere Festlegungen siehe Dokument OCIT-O Protokoll.

Zusätzlich ist eine direkte Abfrage der Gerätezeit durch die Zentrale und Abfrage der Zentralenzeit durch das Feldgerät möglich (siehe Systemobjekt Feldgerät 4.1.1 und Systemobjekt Zentrale 4.1.2). Diese Abfragen sind mit einem durch die Übertragungszeit zwischen Zentrale und Feldgerät bedingten Zeitfehler behaftet.

2.2 Erkennen von Störungen des Übertragungsweges

(Ergänzung durch Definitionen in den Dokumenten OCIT-O Profil_nn möglich)

Unter „Störung des Übertragungsweges“ wird hier ein vollständiger Ausfall der Übertragungsstrecke über mehrere Sekunden, wie er bei einer Unterbrechung der Verbindung oder einem Ausfall der Versorgungsspannung („Netzausfall“) auftreten kann, verstanden. Übertragungsstörungen werden auf defekte Systemteile zurückgeführt. Sporadische Übertragungsstörungen können temporär zu vergleichbaren Störungsbildern führen.

Übertragungsstörungen können verursacht werden durch:

- Ausfall des zentralen Rechners oder des Feldgerätes
- Ausfall von Übertragungseinrichtungen in Feldgerät oder Zentrale
- Unterbrechung des Übertragungsweges
- Netzausfall

Derartige Übertragungsstörungen werden durch das Fehlen von Telegrammen erkannt, wobei die Schnelligkeit der Erkennung abhängig ist von der Häufigkeit der abgeschickten Telegramme:

- **Erkennungsmöglichkeit in der Zentrale:**
  Auf die Telegramme der Zentrale erfolgt keine Antwort.

- **Erkennungsmöglichkeit im Feldgerät:**
  Es treffen keine Telegramme der Zentrale ein.

In den OCIT-Outstations Definitionen sind für Schaltbefehle Start- und Endezeiten vorgesehen, die das Feldgerät bei fehlenden Telegrammen veranlassen, in Vorzugsprogramme zu schalten. Darüber hinaus können in regelmäßigen Abständen Kontrolltelegramme gesendet werden, bei deren ausbleiben auf eine Übertragungsstörung geschlossen werden kann.

Erkennungsmöglichkeiten, die auf Funktionen des Übertragungsgerätes beruhen, wie z.B. Trägerüberwachung, sind hier nicht aufgeführt, da sie typabhängig sind und die Übertragungsgeräte in OCIT-Outstations nicht vorgeschrieben werden.

2.2.1 Unterscheiden von Übertragungsstörungen

Vom Feldgerät erkannte Übertragungsstörungen erzeugen die OCIT-Outstations Meldung **Kommunikationsstörung** (4.2.12) und gelangen über die Abfrage des Standard-Meldearchivs (4.2.9) zur Zentrale.

Die OCIT-Outstations Meldung „Kommunikationsstörung“ schließt die Fehlerursache „Netzausfall“ mit ein. Es besteht aber die Möglichkeit, die Fehlerursache „Netzausfall“ von den anderen Ursachen zu unterscheiden.

2.2.1.1 Unterscheidung nach Beseitigen der Störung

Nach Beseitigen der Kommunikationsstörung werden bei ursächlichem Netzausfall OCIT-Outstations Meldungen aus dem Feldgerät zur Zentrale übertragen (Standard-Meldearchiv), die der Zentrale eine Eingrenzung der Ursache im Nachhinein erlauben:

- **Netz aus** mit Angabe des Ausfallzeitpunkts (4.2.12)

- **Netz ein** (4.2.12)

- **Kommunikation ok** (4.2.12)

Die Zentrale kann damit im Nachhinein den ursächlichen Netzausfall erkennen und die ursprüngliche Meldung „Kommunikationsstörung“ eingrenzen.

2.2.1.2 Unterscheidung sofort nach Auftreten

Diese Option erlaubt den Feldgeräten die sofortige Meldung eines Netzausfalls und damit der Zentrale eine sofortige Eingrenzung der Meldung „Kommunikationsstörung“ auf den vorliegenden Fehlerfall.

Dazu benötigen die Feldgeräte eine Pufferung der Versorgungsspannung (Kurzzeit-USV), um die für die Meldung benötigten Geräteteile über die notwendige Zeit der Melderoutine weiter mit Spannung zu versorgen. Es werden zwei Methoden zur Übermittlung der Information über den Netzausfall definiert, die sich in der Länge der notwendigen Pufferzeit der Versorgungsspannung unterscheiden.

Variante a) Netzausfall-Meldung über Standard-Meldearchiv

> Meldung **Netz aus** mit Angabe des Ausfallzeitpunkts in das Standard-Meldearchiv. Um die Meldung aus dem Standard-Meldearchiv abzuholen sind mehrere Transaktionen notwendig.

Variante b) Netzausfall Meldung über Eventliste

> Bei einem Netzausfall ruft das Feldgerät die Methode **EvListe::OnNetzAus()** in der für das Standard-Meldearchiv registrierten Event-Destination auf. Der Methodenaufruf erfolgt auf dem Kanal btpplHi, um die Übertragung zu priorisieren. In den Parametern der Methode wird die OCIT-Adresse des Feldgerätes und eine Vorgangskennung der Störung übertragen. Neben dem Aufruf der Methode erfolgt eine Protokollierung der Meldung NetzAus im Standard-Meldearchiv, die entweder sofort oder nach Wiederkehr der Versorgungsspannung übertragen wird. Es wird dieselbe Vorgangskennung verwendet, die mit **EvListe::OnNetzAus()** übertragen wird. Dies ermöglicht der Zentrale eine eindeutige Zuordnung von Event und Meldung.

Es bleibt den Herstellern überlassen, ob sie dieses optionale Ausstattungsmerkmal „Erkennen von Netzausfall sofort nach Auftreten“ anbieten oder nicht. Wird diese Option angeboten, gelten folgende Festlegungen:

- Das Feldgerät ist in der Lage, nach Ausfall der Betriebsspannung noch mehrere Sekunden die OCIT-Outstations Kommunikation fortzuführen und die entsprechende Meldung abzusetzen. Die geschätzten Pufferzeiten liegen bei Festverbindungen für Variante a) um um 30 sec, für Variante b) um 5 sec. Bei Wählverbindungen liegt die geschätzte benötigte Pufferzeit bei mindestens einer Minute für beide Varianten. Die Pufferzeiten der Versorgungspannung sind jedoch in OCIT-Outstations nicht vorgegeben.

- Andere Lösungsmöglichkeiten sind nicht OCIT-Outstations konform und nur herstellerspezifisch wirksam.

2.3 Vorgangskennung, SYSJOBID

Im Gesamtsystem werden Bedienvorgänge von verschiedenen Quellen aus durchgeführt (manuell oder automatisch). Diese sind z.B. bei Lichtsignalsteuergeräten eine zentrale Bedienerschaltung einer Gruppe, eine Zeitautomatik Schaltung oder eine VA-Schaltung durch eine lokale verkehrsabhängige Logik (VA). Diese Vorgänge haben eventuell Auswirkungen auf mehrere Geräte. Damit eine Kontrolle der Vorgänge anhand einzelner Meldungen möglich ist, werden Vorgangskennungen eingeführt. Alle Meldungen mit der gleichen Vorgangskennung innerhalb einiger Tage gehören zu dem gleichen Vorgang.

Dazu ist es nötig, dass jeder Vorgangsinitiator eine eindeutige Nummer verwendet. OCIT-Outstations verwendet fest vergebene Nummernbänder. Die Vorgangskennung ist ein Be-

standteil ausgewählter Bedienvorgänge. Sie besteht aus Herkunftskennung und Auftragsnummer. Die Herkunftskennung ist eine eindeutige Kennzeichnung eines Systemteils, der einen Bedienvorgang auslöst. Die Auftragsnummer ist eine für Bedienvorgänge fortlaufend vergebene Nummer.

Auf Grund der Vorgangskennung kann

- das Feldgerät die internen Abläufe ordnen und
- die Zentrale die Bedienvorgänge verschiedener Bediener und Systeme dokumentieren.

Die Vorgangskennung muss während der maximal zu erwartenden Laufzeit eines Bedienkommandos eindeutig sein.

Meldungen des Feldgerätes, die auf Grund von Bedien- oder Änderungsvorgängen entstehen, übernehmen die Vorgangskennung der auslösenden Aktion. Damit können in der Zentrale Vorgang und Reaktion dokumentiert werden.

Die Auftragsnummer vergibt immer der durch die Herkunftskennung bestimmte Systemteil.

Welche Vorgänge die Vorgangskennung mitführen, wird zusammen mit der Definition der Schnittstellenfunktion festgelegt.

Vorgangskennung:

Vorgangskennung					Bezeichnung
Herkunft				Auftragsnummer	
Systembezeichnung	Verursacher				
Teilsystem	Typ	Untertyp	Instanz		
1 = Zentrale 2 = Systemzugang 3 = Feldgerät	0 = keine Detaillierung	-	0 bis 63 0 bis 63 0 bis 65535	0 bis 65535 0 bis 65535 0 bis 63	
	1 = ZAUT	0 = keine Detaillierung 1 -15 Nr. der Zeitautomatik			Zeitautomatik/Schaltuhr
	2 = VA-Logiken	0 = keine Detaillierung 1 - 15 Nr. der Logik			Verkehrsabhängige Logiken oder Anwenderprogramme im Feldgerät/Zentrale

Vorgangskennung					Bezeichnung
Herkunft				Auftrags-nummer	
Systembe-zeichnung	Verursacher				
Teilsystem	Typ	Untertyp	Instanz		
	3 = Be-dienzu-gänge	0 = keine Detaillierung 1 = Bediengerät integriert 2 = Bediengerät abgesetzt 3 = Service PC 4 = Sondereingriff 5 - 15 weitere Bedienzugänge			Bedienzugänge über lokale und zentrale Be-diengeräte / PCs
	4 = Über-tragungs-systeme	0 = keine Detaillierung 1-15 für genauere Angabe des Übertragungssystemteils			
	5 = Ü-berwa-chungen	0 = keine Detaillierung 1 -15 für genauere Angabe der Überwachungs-einrichtung			
	6 - 15 frei				

Teilsystemkennungen:

Kenn Nr.	**Teilsystem**
0	nicht festgelegt
1	Zentrale
2	Systemzugang
3	Feldgerät

Formate:

2^{32}				2^0
Teilsystem	Typ	Untertyp	Instanz	Auftragsnummer
2 bit	4 bit	4 bit	Zentrale: 6 bit	Zentrale: 16 bit
			Systemzugang: 6 bit	Systemzugang: 16 bit
			Feldgerät: 16 bit	Feldgerät: 6 bit

Beispiele:

Bedienungsvorgang von zentraler Zeitautomatik (ZAUT), Zentrale 0:

1	1	1	0	Auftrags-nummer	JAUT Tagesplan

Bedienungsvorgang über zentrale Systemkomponente, Zentrale 0:

1	3	1	0	Auftrags-nummer	Zentrale Bedienung (manuell)

Bedienung über lokales, abgesetztes Bediengerät (z.B. Handpanel), Feldgerät 317:

3	3	2	317	Auftrags-nummer	Abgesetztes Be-diengerät

2.4 Zählweise bei nummerierten Elementen

- 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.

- Die Zählung von Zeiten beginnt mit Zeit 0. Zeit 0 bezeichnet den ersten Zeittakt von seinem Beginn bis zum seinem Ende.

3 Systemzugänge

In einem OCIT-Outstations-System sind folgende Systemzugänge vorgesehen:

- Zentraler Systemzugang
  Erlaubt den Anschluss von Servicetools in der Zentrale und ermöglicht darüber den Zugang zu den Feldgeräten. Für die Anwendung „Zentraler Systemzugang“ wird das zentrale LAN verwendet und die Schichten 2 und 1 entsprechend angepasst.

- Lokaler Systemzugang
  Erlaubt den Anschluss von Servicetools am Steuergerät und ermöglicht darüber den Zugang zur Zentrale und anderen Feldgeräten. Für die Anwendung „Lokaler Systemzugang“ sind derzeit noch keine Festlegungen getroffen.

3.1 Zentraler Systemzugang

Der Systemzugang in der Zentrale besteht aus einer oder mehreren Schnittstellen, die eine Kommunikation mit den Feldgeräten erlauben. Es handelt sich um logisch völlig identische Schnittstellen, wie sie auch zu den Feldgeräten führen, die Anbindung erfolgt jedoch über LAN. Eine OCIT-Zentrale muss mindestens einen Systemzugang bieten:

Schnittstellen	Verbindlich	projektspezifisch / herstellerspezifisch
Anzahl	1	> 1
Übertragungsprofil	LAN 10 Mbps / LAN 100 Mbps 10/100 Base T Ethernet Stecker RJ-45	ISDN oder andere Dienste
Protokoll	OCIT wie zu den Feldgeräten	

Die Hersteller können jedoch zusätzlich mehrere Zugänge vorsehen und auch ISDN-Anschlüsse oder andere Dienste bieten.

Der Betreiber / Zentralenhersteller stellt pro zentralen Systemzugang folgende Informationen zur Verfügung:

- IP Adresse des anzuschließenden Systemzugangsrechners
- IP Adresse des Gatewayrechners (falls nötig)
- IP Adresse des Namensservers (DNS)
- Vom Systemzugang zu verwendende OCIT Zentralennummer, OCIT Feldgerätenummer.

Der Systemzugang ist in erster Linie für Experten gedacht, die darüber von der Zentrale oder von entfernten Orten aus z.B. die Geräteversorgung ihrer firmeneigenen Geräte durchführen

oder Gerätefunktionen testen können. Ein Feldgerät führt alle Kommandos des zugeschalteten Servicetools aus. Fehlbedienungen können die Systemfunktion stören. Zugang zu den Geräten erhalten daher nur autorisierte Nutzer, die ihre Berechtigung durch die Kenntnis von Passwörtern nachweisen müssen.

Prinzipiell ist es möglich, über den Systemzugang auch Schaltwünsche abzusetzen, welche den Ist-Zustand des Geräts beeinflussen. Dies kann im Konflikt zu den Schaltwünschen der Zentrale stehen. Da der Systemzugang ein Expertenzugang ist und demzufolge erwartet wird, dass die Nutzer wissen was sie tun wird hier folgendes Verhalten definiert:

> Ein gültiger Schaltwunsch wird so akzeptiert, wie wenn er von der Zentrale käme und demzufolge auch entsprechend angesteuert. In der Zentrale wird die Zustandsänderung über den Ist-Vektor sichtbar und über die SysJobId kann der Verursacher ermittelt werden. Ein erneuter Befehl der Zentrale überschreibt dann den vorhergehenden Schaltwunsch erneut. Es gilt das Prinzip Last come – first serve.

3.2 Lokaler Systemzugang

Für die Anwendung „Lokaler Systemzugang“ sind derzeit noch keine Festlegungen getroffen.

4 Objektdefinitionen

Datendefinitionen siehe OCIT-O-Basis-TYPE_Vy.y.xml.

4.1 Systemobjekte

OType Nummern der Systemobjekte, Member=0 (OCIT-Outstations):

OType	Name	Pfad (ab Feldgerät)
815	Systemobjekt Feldgerät	./.
817	Systemobjekt RemoteDevice	ZNr(USHORT)/FNr(USHORT)

4.1.1 Systemobjekt Feldgerät

Dieses Objekt dient dazu, allgemeine Informationen über das Feldgerät zur Verfügung zu stellen. Außerdem erlaubt es dem Feldgerät, neue Kommunikationspartner bekannt zu machen bzw. deren Passwörter zu ändern.

SystemobjektFeldgeraet		
METHOD	**Name**	**Beschreibung**
100	**GetGeraeteID**	Liest den Hersteller,Version und Gerätetyp aus. Diese Parameter können sich bei Updates ändern.
	Ausgabeparameter	
	RetCode : RetCode	OK: Funktion wurde korrekt durchgeführt.
	FgType	1: Zentrale 2: Systemzugang 3: Feldgerät
	Member : MemberID	ID des Geräteherstellers siehe OCIT-O Protokoll
	Devicetype : string	Typ des Geräts, welches angeschlossen ist s. Geräte-TYPE-Datei, Tag OCT.DEVICETYPE.
	Version : string	OCIT-Version, s.a. Geräte-TYPE-Datei, Tag OCT.VERSION
	SubVersion : string	Herstellerspezifische Versionskennung, z.B. Softwareversion, s.a. Geräte-TYPE-Datei, Tag OCT.SUBVERSION
	APVersion : string	Version der Anwenderprogramm Software
101	**CreateRemoteEntry**	legt einen neuen Remote-Eintrag an
	Eingabeparameter	
	ZNr, FNr	ZNr, FNr des fremden Gerätes
	RemoteType	1: Zentrale 2: Systemzugang 3: Feldgerät
	Ausgabeparameter	
	RetCode : RetCode	OK: Entry wurde angelegt. TOO_MANY: Entry-Tabelle ist voll. EXISTS_ALREADY: RemoteDevice existiert bereits.
102	**DropRemoteEntry**	löscht den Remote-Eintrag
	Eingabeparameter	
	ZNr, FNr	ZNr, FNr des fremden Gerätes
	Ausgabeparameter	
	RetCode : RetCode	OK: Entry wurde angelegt. PARAM_INVALID falls zu löschender Eintrag nicht existiert.
103	**GetTime**	gibt aktuelle Zeit des Geräts zurück
	Ausgabeparameter	
	RetCode : RetCode	OK: Zeit wurde geliefert.

SystemobjektFeldgeraet		
METHOD	**Name**	**Beschreibung**
	zeit : ZEITSTEMPEL_UTC	aktuelle Zeit des Gerätes
	ZEITZONE : SLONG	Zeitzone (Abweichung in Sekunden von GMT)
	ZEITQUELLE : ENUM	Zeitquelle (unbekannt, Quarz, Zentrale, DCF, GPS)
104	**InstanceInfo**	Liefert alle Instanzreferenzen die zu der im folgenden angegebenen Referenz passen. Der Pfad kann auch nur teilweise angegeben werden. Die Methode liefert dann alle Referenzen auf Instanzen welche vom angegebenen Typ oder einer Spezialisierung davon sind und deren Pfad wie der angegebene Pfad beginnt. Diese Methode kann z.B. dazu benützt werden, alle Aufträge zu einer Liste auszulesen.
	Eingabeparameter	
	key : BaseObjType?^3	key.RefLen key.Member key.Otype Member, OType gibt den Basisdatentyp an, von welchem die Methode Instanzreferenzen liefert. key.path .. Pfadparameter je nach oben angegebenem Typ. Vom Ende des Pfades können Parameter weggelassen werden.
	Ausgabeparameter	
	RetCode	OK Funktion wurde korrekt durchgeführt (auch wenn Klasse bekannt aber keine Instanz gefunden). PARAM_INVALID falls die durch key referenzierte Klasse nicht bekannt ist.
	Path[] : paths	Instanzreferenzliste bestehend aus: paths.Anzahl paths[].RefLen paths[].Member paths[].Otype paths[]. ... Pfadparameter je nach zur Laufzeit angegebenem Typ.

Die von der Methode GetGeraeteID gelieferten Versionskennungen erlauben eine Abfrage des unterstützten Funktionsumfangs des Gerätes. Insbesondere das Element VERSION erlaubt es z.B. einer Zentrale, den unterstützten Versionsumfang eines Feldgerätes entsprechend seinem OCIT-Versionsstand auszulesen und dementsprechend zu bedienen.

Die OCIT Versionskennung besitzt das Format *Version.Subversion*, also

OCIT-Instations V1.0	„1.0“ oder „1“ (Abwärtskompatibilität)
OCIT-Instations V1.1	„1.1“
OCIT-Instations V2	„2.0“

4.1.2 Systemobjekt Zentrale

Da die Zentrale aus BTPPL-Sicht auch ein Feldgerät ist, existiert das Systemobjekt Feldgerät auch in der Zentrale.

4.1.3 Systemobjekt Remote Device

Die Remote Device Objekte stellen individuelle Passwörter für den Zugriff von anderen IP-Adressen bzw. anderen Feldgeräten zur Verfügung. Mit diesem Objekt kann das Passwort geändert werden. Die Instanz, deren ZNr und FNr gleich der des enthaltenden Feldgerätes sind, beschreibt das Default Passwort für unbekannte IP-Adressen.

RemoteDevice		
Pfad: ZNr, FNr		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	IpAdresse	Eingestellte IP-Adresse dieses Remote Devices
	IpName	IP Hostname dieses Remote Devices
	FgTyp	1: Zentrale 2: Systemzugang 3: Feldgerät
100	**SetPassword**	Feldgeräte kennen mindestens 4 Passwörter: - Zentrale - Ersatzzentrale - Systemzugang - Default Jedem Passwort außer Default sind IP-Adresse, ZNr, FNr und Hostname des zugelassenen Kommunikationspartners zugeordnet. Abhängig von der Absender-IP-Adresse ermittelt BTPPL das passende Passwort.
	Eingabeparameter	

RemoteDevice		

	NewPassword : UBYTE[20]	Das Array ist folgendermaßen aufgebaut:
		`NewPassword[0..11]=neues OCIT-Passwort[0..11] XOR Schleier[0..11]`
		`NewPassword[12..19]=Schleier[12..19]`
		Wenn das OCIT-Passwort kleiner als 12 Zeichen ist, wird es mit binären 0en aufgefüllt. Das Passwort wird es mit einem Schleier durch eine XOR-Operation unkenntlich gemacht.
		Der Schleier wird mit Hilfe des SHA-1 Algorithmus gebildet. Hierzu wird mit dem alten OCIT-Passwort des Gerätes folgende Formel ausgeführt:
		Schleier := SHA-1(altes OCIT-Passwort + ´.´ + ZNRZieladresse + ´.´ + FNRZieladresse + SCHLEIER + altes OCIT-Passwort + ´.´ + ZNRZieladresse + ´.´ + FNRZieladresse)
		Der SCHLEIER ist ein hexadezimal definiertes Array mit folgendem Inhalt:
		00h: 49 61 65 21 20 49 61 65 21 20 50 68 20 6E 67 6C
		10h: 75 69 20 6D 67 6C 77 20 6E 61 66 68 20 43 74 68
		20h: 75 6C 68 75 20 52 20 6C 79 65 68 20 77 61 67 6E
		30h: 20 6E 61 67 6C 20 66 68 74 61 67 6E
		Beispiel: Wenn das alte OCIT-Passwort 'OCITPASSWORT' und das Feldgerät an der Zentrale 12 mit der Gerätenummer 567 hängt, ist der Schleier die SHA-1 Prüfsumme von der Zeichenfolge
		00h: 4F 43 49 54 50 41 53 53 57 4F 52 44 2E 31 32 2E ; OCITPASSWORD.12.
		10h: 35 36 37 49 61 65 21 20 49 61 65 21 20 50 68 20 ; 567
		20h: 6E 67 6C 75 69 20 6D 67 6C 77 20 6E 61 66 68 20
		30h: 43 74 68 75 6C 68 75 20 52 20 6C 79 65 68 20 77
		40h: 61 67 6E 20 6E 61 67 6C 20 66 68 74 61 67 6E 4F ; O
		50h: 43 49 54 50 41 53 53 57 4F 52 44 2E 31 32 2E 35 ; CITPASSWORD.12.5
		60h: 36 37 ; 67
		Dadurch ändert sich der Schleier je nach Feldgerät, selbst wenn das Passwort für alle Geräte gleich ist.
		Das Feldgerät berechnet denselben Schleier (mit seinem alten Passwort). Wenn dieses wieder über den Text gelegt wird, kommt das neue Passwort wieder zum Vorschein. Es wird dann bei allen weiteren Operationen eingesetzt.

	Ausgabeparameter	
	RetCode : RetCode	OK: Passwort geändert. NO_CENTRAL: Zentralenpasswörter dürfen nur durch die Zentrale geändert werden.

4.1.4 RemoteService

Es gibt mehrere Quellen für steuernde Befehle, wobei jede Quelle ihre eigene Priorität hat:

1. Bedienung oder Service „lokal“
2. Bedienung oder Service „remote“
3. Zentrale
4. lokale Kommandowahl, z.B. Zeitautomatik (niedrigste Priorität)

Gedanklich entspricht ein Remote-Serviceeingriff einem verlängerten Handgerät.

Im Unterschied zu lokalen Serviceeingriffen kann das Gerät bei Remote-Serviceeingriffen das Ende nicht über den Türkontakt feststellen. Deshalb sieht OCIT einen zeitlich befristeten Befehl zum Einstellen des Servicebetriebs vor.

RemoteService **Pfad: ./. (nur eine Instanz pro Feldgerät)**		
METHOD	**Name**	**Beschreibung**
0	**Get**	
	Ausgabeparameter	
	EndZeit :ZEITSTEMPEL_UTC	Zeit bis wann die Remote Service Bedienung aktiv ist. EndZeit==0 falls keine Servicebedienung aktiv ist.
	VorgangsNr : SYSJOBID	Kennung des Vorgangs welcher die Remote Service Bedienung angefordert hat, oder NULL falls keine Remote Service Bedienung aktiv ist.
	ServiceGrund : string	Gibt an warum Service Bedienung angefordert wurde.
100	**StartService**	Damit kann ein Service PC die Gerätebedienung anfordern. Gerät darf zu einer Zeit die Bedienberechtigung nur einmal erteilen. Falls die Remote Service Bedienung schon aktiv ist, liefert die Methode ACCESS_DENIED zurück.
	Eingabeparameter	
	EndZeit :ZEITSTEMPEL_UTC	Zeit bis wann die Remote Service Bedienung aktiv ist. EndZeit==0 falls keine Servicebedienung aktiv ist.
	VorgangsNr : SYSJOBID	Kennung des Vorgangs welcher die Remote Service Bedienung angefordert hat, oder NULL falls keine Remote Service Bedienung aktiv ist.
	ServiceGrund : string	Gibt an warum Service Bedienung angefordert wurde.
	Ausgabeparameter	
	RetCode : RetCode	OK: Entry wurde angelegt. PARAM_INVALID, INTERVALL_INVALID,ACCESS_DENIED

RemoteService Pfad: ./. (nur eine Instanz pro Feldgerät)		
METHOD	Name	Beschreibung
101	**EndService**	Mit EndService gibt ein Service PC das Bedienrecht wieder zurück. Danach berücksichtigt das Gerät Zentralenschaltwünsche wieder. In Ausnahmefällen kann es nötig sein dieses Kommando auch von der Zentrale abzusetzen.
	Ausgabeparameter	
	RetCode : RetCode	OK, (auch wenn Remote Service Bedienung nicht aktiv war)ACCESS_DENIED

4.1.4.1 Direkter externer Zugang zum Feldgerät

Direkte externe Zugänge zu den Feldgeräten über Wählverbindungen/Netzwerke oder die lokale Serviceschnittstelle liegen in der Verantwortung des Feldgeräteherstellers. Sie werden projekt- bzw. herstellerspezifisch realisiert.

Falls zum Feldgerät neben der Verbindung zur Zentrale ein solcher direkter externer Zugang hergestellt wird, ist von OCIT-konformen Geräten folgendes zwingend zu erfüllen:

Beginn des Zugangs: Meldung WARTUNG_EIN zur Zentrale.

Ende des Zugangs: Meldung WARTUNG_AUS zur Zentrale.

Bei direkten Wählverbindungen kann es dabei möglich sein, dass die Zentrale für die Zeit eines direkten Zuganges zum Feldgerät abgetrennt wird, und daher eine Verbindungsstörung erkennt. Dieser Störungszustand kann nach dem Beenden des Zugangs korrigiert werden, da eine entsprechende Meldung vom Feldgerät zur Zentrale erfolgt.

4.2 Meldungen und Messwerte (Archive)

In OCIT-Outstations werden Meldungen und Messwerte in Archiven der Feldgeräte abgelegt. Die Methoden, mit denen die Zentrale diese Archive abfragt, sind für Meldungen und Messwerte gleich. OCIT-Outstations vereinigt Messwert- und Meldungsarchive unter einer gemeinsamen Schnittstelle.

4.2.1 Eigenschaften der Archive

In Archiven werden ausgewählte Daten und Meldungen des Feldgerätes gesammelt. Hierfür stellt OCIT folgende Elemente zur Verfügung (Details siehe Pkt. 4.2.9):

- Eine allgemeine Archivschnittstelle, welche die einfache Verwaltung von Archiven zulässt,
- Standard- und optionale Archive für Meldungen,

- zur Laufzeit von der Zentrale definierbare Messwertarchive. Damit für Meldungen eine definierte Speichertiefe eingehalten wird, sieht OCIT-Outstations für Messwerte eigene Archive vor.

Die Archivschnittstelle für Meldungen und Messwerte hat folgende Vorteile:

- Es lassen sich Messwerte hinzufügen, ohne dass in OCIT-Outstations Erweiterungen gemacht werden müssen.
- Die Daten werden kompakt übertragen.
- Es ist möglich, herstellerabhängig zusätzliche Archive zu definieren.
- Messwerte können mehrfach abgeholt bzw. archiviert werden.
- Die möglichen, eintragbaren Meldungen im Archiv sind erweiterbar.
- Es ist möglich, herstellerabhängig einzelne Meldungen (durch neue Meldungsteile) zu erweitern.
- Trotz der Erweiterung einer Meldung durch neue Meldungsteile lässt sich die ursprüngliche Meldung einfach herausfiltern.
- Durch Updates zur Zentrale bei neuen Meldungen bleibt die Zentrale auf dem neuesten Stand.
- Zentrale kann das gesamte, noch vorhandene Archiv lesen.
- Möglich ist eine inkrementelle Übertragung zu Zentrale1 parallel zu Gesamtlesen von Zentrale 2 (z.B. Systemzugang).
- Datenverlust erst bei Überlauf des Zwischenspeichers, nicht wegen Übertragungsstörung allein.
- Die Zentrale kann auf den jüngsten Eintrag zugreifen (für aktuellen Status bei Verbindungsaufbau).
- Die Meldungen lassen sich projektspezifisch auf Kundenwunsch ohne Firmwareänderungen anpassen.

4.2.2 Die Archivschnittstelle

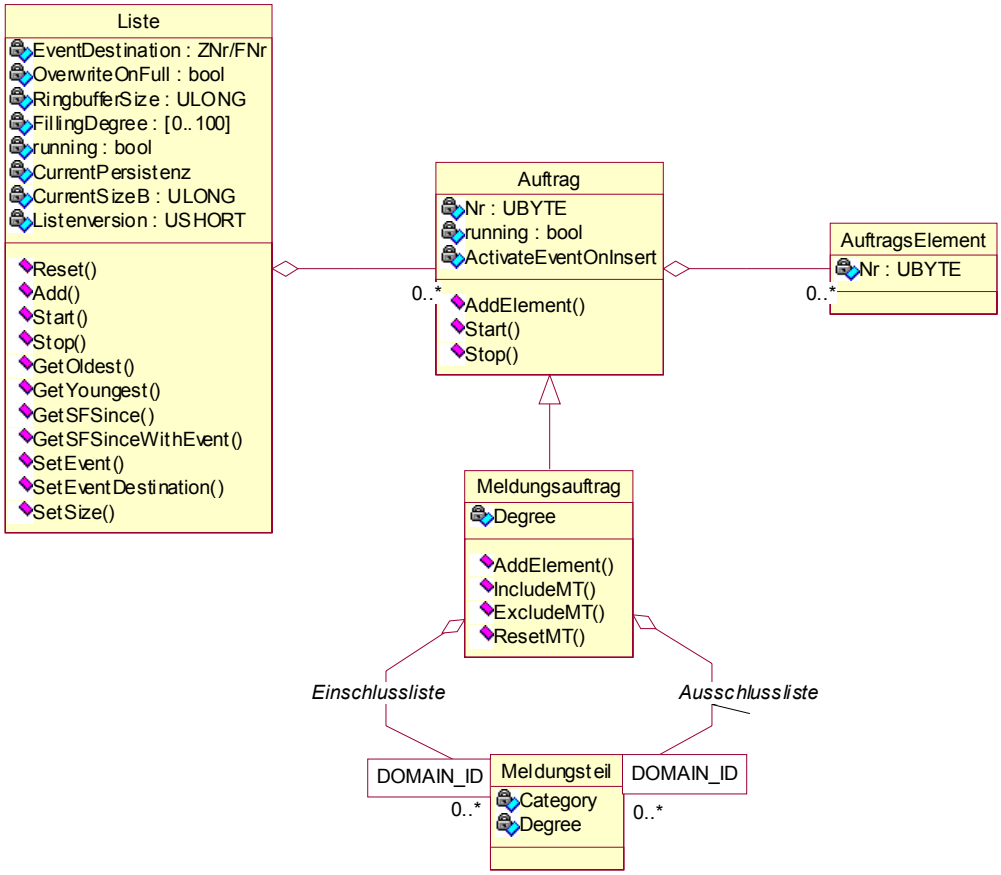
Meldungen und Messwerte werden in einer gemeinsamen Schnittstelle behandelt. Die Datenstrukturen und die definierten Funktionen der Schnittstelle sind für Meldungen und Messwerte strukturell gleich.

Meldungen und Messwerte werden in ‚Listen' abgespeichert. Es gibt im Feldgerät mehrere Listen, die unterschiedliche Daten abspeichern. Welche Daten abgespeichert werden, wird durch ‚Aufträge' festgelegt und in der jeweiligen Liste abgespeichert. Diese Konfiguration kann während des Betriebs ausgelesen werden. Es gibt vordefinierte Listen, die vom Gerätehersteller fest definiert sind und dynamische Listen, die von der Zentrale während des Betriebs konfiguriert werden können. Pro Liste sind bis zu 256 verschiedene Aufträge möglich.

Jede Liste hat einen eigenen Puffer, in dem die dynamischen Daten gespeichert werden. Der Puffer ist als Ringpuffer ausgeführt, in dem die jeweils ältesten Sätze überschrieben werden. Die Größe des Ringpuffers ist konfigurierbar, kann aber während der Messwertaufnahme nicht geändert werden.

Ein Ringpuffer besteht aus ‚Sekundenframes'. Es ist möglich, dass mehrere Sekundenframes innerhalb der gleichen Sekunde angelegt werden. Ein Sekundenframe ist in jedem Fall durch die (sekundengenaue) Zeit gekennzeichnet, für die der Sekundenframe angelegt wird. Es gibt nur für die Zeitpunkte Sekundenframes, für die auch Daten abgespeichert werden. Ein Sekundenframe enthält nur sogenannte ‚Auftragsframes'. Ein Auftragsframe speichert die dynamischen Daten, die durch einen Auftrag gefordert waren. Es gibt für Meldungen und für Messwerte unterschiedliche Strukturen von Auftragsframes. Sowohl Meldungen als auch Messwerte sind vollständig in einem Sekundenframe gespeichert. Pro Sekundenframe sind mehrere Messwerte und Meldungen möglich. Meldungen und Messwerte werden normalerweise in verschiedenen Listen gespeichert.

Der Ablauf sieht folgendermaßen aus: Das Feldgerät speichert neue Daten in seinem Ringpuffer, überträgt die Daten aber nicht automatisch zur Zentrale. Die Daten werden nur auf Anfrage von der Zentrale an diese übertragen. Damit ist es möglich, die Daten sekundengenau abzuspeichern, sie können jedoch in größeren Zeitabständen übertragen werden. Zusätzlich kann die Zentrale (mit Events) benachrichtigt werden, wenn der Puffer einen von der Zentrale festgelegten Füllgrad überschreitet. Zusätzlich können Events ausgelöst werden, wenn bestimmte Ereignisse eintreten (z.B. wenn eine Lampe ausgebrannt ist). Das Abholen der Daten geschieht über BTPPL-Methodencalls, die mindestens zusätzlich zu PPP und TCP über eine 16-Bit Prüfsumme (Fletcher) gesichert sind und optional über eine 160 Bit Prüfsumme (SHA-1) abgesichert werden können.



Schema der Archivschnittstelle

4.2.3 Elemente der Archivschnittstelle

Die Archiv-Schnittstelle ist in folgende abstrakte Elemente untergliedert:

- Eine **Liste** verwaltet Aufträge und die dazugehörigen dynamischen Werte, wie Messwerte oder Meldungen. Welche Werte erfasst und gespeichert werden, wird durch die Aufträge festgelegt. Es gibt statische Listen, die nicht neu definiert werden können und dynamische Listen, die von der Zentrale neu konfiguriert werden können.
- **Aufträge** gehören zu einer Liste und legen fest, welche dynamischen Werte gespeichert werden sollen. Ein Auftrag besteht aus einem oder mehreren Auftragselementen.
- **Auftragselemente** verweisen auf die Objekte (Prozessvariablen) und enthalten die Information, welche Werte dieser Objekte erfasst werden. Im Normalfall verweist ein Typ von Auftragselement genau auf einen Typ von OCIT-Outstations-Objekt, dessen Daten es erfasst. Es ist aber möglich, dass eine Prozessvariable von mehr als einem Auftragselement (in verschiedenen Listen) verwendet wird. Jedes Element in OCIT-Outstations, für das ein Auftragselement vorhanden ist, kann als **Datenquelle** eingesetzt werden.
- Der **Auftragsframe** ist das Ergebnis eines Auftrags. Auch ein Auftrag, der aus mehreren Auftragselementen besteht, erzeugt immer genau einen Auftragsframe.
- Die dynamischen Daten werden in **Sekundenframe**s gespeichert. Ein Sekundenframe enthält z.B. eine Reihe von Messwerten in Form von Auftragsframes, die in der gleichen Sekunde entstanden sind, oder alle Meldungsteile, die zu einer Meldung gehören
- Ein spezieller Auftragsframe ist der **Meldungsauftrag** zur Erfassung von Meldungen. OCIT-konforme Meldungen bestehen aus einem **Hauptmeldungsteil** und 0..n **Zusatzmeldungsteilen** (optionalen Meldungsteilen). Der Hauptmeldungsteil bestimmt die Semantik der Meldung. Eine Meldung wird deshalb in Meldungsteile aufgeteilt, damit verschiedene Hersteller eine bereits vorhandene Standardmeldung erweitern können. Durch die Gruppierung wird dargestellt, dass die Meldungsteile semantisch zusammengehören. Ein **Meldungsteil** besteht aus einer Kennung und einem Satz von Parametern, welcher die Meldung genauer definieren. Die Kennung besteht aus der Herstellerkennung (Member-Nummer)und aus der Typkennung (OType).

OType Nummer der Listen in der Zentrale, Member=0:

OType	**Name**	**Pfad (ab Feldgerät)**
400	Liste	Liste(UBYTE)
401	EvListe	EvListe
402	Auftrag	Liste(UBYTE)/Auftragsnummer(UBYTE)
405	Meldungsauftrag	Liste(UBYTE)/Auftragsnummer(UBYTE)
430	AuftragsElement	Liste(UBYTE)/Auftragsnummer(UBYTE)/AENr(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.

4.2.4 Liste

Eine Liste hat zwei Aspekte: In einem statischen Teil wird abgelegt, welche Aufträge existieren und damit welche dynamischen Daten übertragen werden. Für diese Aufträge gibt es ein **Prüfwort (Listenversion)**, welches bei jeder Änderung der Aufträge neu gesetzt wird und damit die Prüfung einfach macht, ob die übertragenen dynamischen Daten und die Auftragsversorgung kompatibel zueinander sind.

Der dynamische Teil der Liste (Ringpuffer) speichert die angefallenen Daten in sog. „Sekundenframes“. Diese Daten können dann von der Zentrale abgeholt werden. Es ist möglich, die Daten mehr als einmal abzuholen, da sie auch nach dem Abholen weiter gespeichert bleiben. Dadurch können z.B. auch Systemzugänge die aktuellen Messwerte verwenden. Es ist nicht vorgesehen, dass Systemzugänge Aufträge löschen oder sich als Eventziel beim Gerät eintragen (Gerät ist **nicht** Multimasterfähig). Die Daten werden in einem Ringpuffer gespeichert, bei welchem die alten Werte einfach überschrieben werden.

Zusätzlich gibt es die Möglichkeit für die Zentrale, sich einen Event schicken zu lassen, wenn ein vorher parametrierter Füllgrad erreicht ist. Zusätzlich kann angegeben werden, wie viel Platz für den Ringpuffer einer Liste verwendet werden soll.

Die Listen sind fest adressiert (Pfad absolut zum Gerät, unabhängig von relativem Knoten) und übertragen Daten übergreifend zum relativen Knoten.

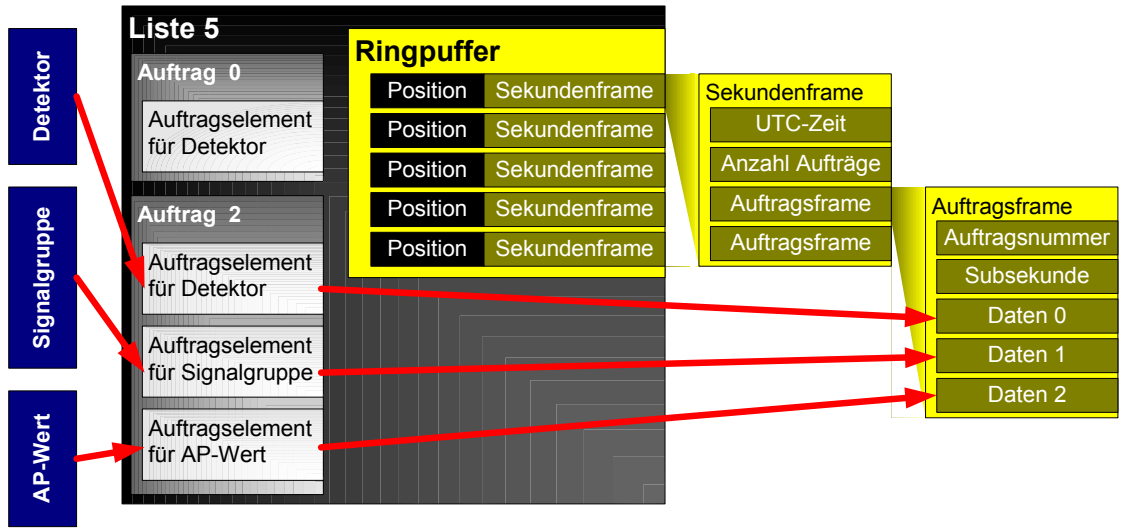
Jeder Sekundenframe hat einen Zeitstempel (sekundengenau) und eine Positionsnummer. Die Positionsnummer ist notwendig, um mehrere Elemente mit gleichem Zeitstempel voneinander zu unterscheiden. Die gleiche Positionsnummer darf im Ringpuffer der gleichen Liste nicht zweimal auftreten, kann aber „lückenhaft“ sein (auf 10 kann z.B. 50 folgen). Es ist im Gerät nicht notwendig, die Positionsnummern mit abzuspeichern. Stattdessen kann auch eine 32-Bit Speicheradresse oder eine Dateiposition verwendet werden. Der Ringpuffer muss selbstverständlich so groß sein, dass er mehr Elemente aufnehmen kann, als pro Sekunde maximal erzeugt werden können. (Das Tupel Zeit/Position darf pro Liste nur einmal vorkommen).

Pro Liste kann festgelegt werden, ob die Abfragen gesichert erfolgen müssen oder ungesichert sein können. Dies ist notwendig, da zum einen bei sehr großen Datenmengen (Onlinemesswerte) eine gesicherte Übertragung Systemressourcen benötigt, zum anderen aber bestimmte Listen sicherheitsrelevant sind.

Zusammenfassung:

- Das Prüfwort ermöglicht Konsistenzprüfung zwischen Listenkonfiguration und abgeholten Daten.
- Die Listeninstanzen sind vordefiniert, aber nicht aktiviert.
- Die Listen sind fest adressierbar (Pfad absolut zum Gerät, unabhängig von relativem Knoten).
- Die Liste verwaltet Datenframes.
- Die Daten werden über das Interface der Liste abgeholt.
- Zur Liste gehört ein Füllgrad, bei dem - optional - die Liste einen Event zur Zentrale schickt (Füllgrad in %).
- Die Liste hat eine feste Eigenschaft, wie viel Platz für die Frames reserviert ist.
- Festlegbar ist das Verhalten bei Überfüllung (Stoppen oderÜberschreiben).

- Die maximale Größe des Ringpuffers ist Eigenschaft der Liste. Die Liste bzw. die zugehörigen Aufträge verhindern die Definition von Aufträgen, deren Sekundenframes größer sind als der Ringpuffer.
- Jede Liste besitzt einen Ringpuffer von Sekundenframes, der abgefragt werden kann. Ein abgefragter Sekundenframe wird nicht gelöscht, sondern verbleibt im Ringpuffer, bis er überschrieben wird.
- Jeder Sekundenframe hat einen Zeitstempel (sekundengenau) und eine (Ringpufferweit eindeutige) Positionsnummer.
- In einem Sekundenframe sind die Daten von einem oder mehreren Aufträgen gespeichert.
- Es ist möglich, dass mehrere Sekundenframes für die gleiche Sekunde existieren. Die Daten für einen Auftrag innerhalb derselben Sekunde sind immer innerhalb des gleichen Frames.
- Es ist möglich, dass in einem Sekundenframe die gleiche Auftragsnummer mehrfach auftritt (speziell bei Meldungen).
- Pro Liste kann festgelegt werden, ob die Abfragen gesichert erfolgen müssen oder ungesichert sein können.
- Listen können in **einem** anderen Feldgerät (meist der Zentrale) Events auslösen. Ein Event ist ein Methodenaufruf des Objekts EvListe. Ein Event wird ausgelöst, wenn sich der Zustand der Liste signifikant geändert hat.



Schema einer Liste

4.2.4.1 Übergabeformat

Das Feldgerät speichert nur Elemente chronologisch in dem Ringpuffer jeder Liste. Es speichert nicht, ob die Elemente schon von der Zentrale gelesen wurden.

Jeder Eintrag im Ringpuffer (Sekundenframe) hat einen Zeitstempel (UTC), also mit Sekundenauflösung. Es ist möglich, dass innerhalb einer Liste mehrere Elemente mit dem gleichen Zeitstempel existieren. Diese können im Fall einer Uhrzeitkorrektur auch Rückwärtssprünge aufweisen. Uhrzeitkorrekturen meldet das Gerät mit der Betriebsmeldung (4.2.12) Zeitsprung. Damit kann die Zentrale den chronologischen Zusammenhang der Daten wiederherstellen.

Jeder Sekundenframe besteht aus einer Liste von Auftragsframes. Der Aufbau eines Auftragsframes hängt vom Typ (Member, OType) des Auftragsframes ab (Achtung: Diese Member/OType Kombination des Auftragsframes ist nicht die Meldungsnummer und wird nicht übertragen). Auftragsframes von Meldungen enthalten variable Parameter und unter anderem die Member/OType Kombination der Meldung (die auch übertragen wird).

Innerhalb einer Liste ist jeder Sekundenframe durch den Zeitstempel und eine Positionsnummer (innerhalb des Ringpuffers) eindeutig identifizierbar. Die Positionsnummer muss innerhalb des Ringpuffers eindeutig sein, sie kann aber Lücken haben. Denkbar ist als Positionsnummer beispielsweise ein File-Offset innerhalb einer Datei im Feldgerät oder die Speicheradresse des Elements im Speicher. Damit dies möglich ist, wird die Positionsnummer als ui4 festgelegt. Die Positionsnummer darf nur den Wertebereich 0x0 .. 0xfffffffe umfassen, also kein 0xffffffff enthalten.

Wenn eine Liste von Sekundenframes vom Feldgerät in die Zentrale übertragen wird, überträgt man zusätzlich den Zeitstempel und die Positionsnummer des vorherigen Elements und den Zeitstempel und die Positionsnummer des letzten übertragenen Sekundenframes.

4.2.4.2 Methoden für Liste

Generell werden alle Methoden, die auf die Liste ausgeführt werden, authentifiziert. Die einzige Ausnahme ist GetSFSince, da GetSFSince keine Veränderung in der Liste selbst vornimmt. GetSFSinceEvent wird immer authentifiziert übertragen, bei der Antwort kann aber auf eine Authentifizierung verzichtet werden, sofern AuthenticateAnswer auf 0 gesetzt ist.

> Faustregel: Wichtige Daten wie Betriebsmeldungen werden authentifiziert; große Datenmengen, die nicht „gerichtsrelevant“ sind, wie Messwerte, werden zu Gunsten der Performance nicht authentifiziert.

In jeder Liste existiert implizit ein Auftrag mit der Nummer 0. Dieser fügt die folgenden Ereignisse als Meldungen in den Ringpuffer der Liste ein:

Suspend, Unsuspend, Zeitsprung, StartAuftrag, StopAuftrag.

Interface des OBJTYPE Liste Member =0, OCIT Standard OType = 400 Liste

Liste		
METHOD	**Name**	**Beschreibung**
100	**GetOldest**	Ältestes Listenelement lesen
	Ausgabeparameter	
	RetCode : RetCode	OK: ältester Sekundenframe(SF) richtig geliefert NO_SF: Liste enthält überhaupt keinen Sekundenframe
	PosNr : ULONG	Positionsnummer des gelieferten SF

Liste		
METHOD	**Name**	**Beschreibung**
	Listenversion: ui2	Versionskennung (Prüfwort), die bei jeder Auftragsänderung vom Feldgerät geändert wird. Die Versionsnummer wird auch beim Start der Liste zurückgegeben, so dass die Zentrale feststellen kann, ob eine Änderung stattgefunden hat. Wie unmittelbar aus den Datenstrukturen ersichtlich ist, ist die Versionsnummer der Liste nicht die Versionsnummer des Sekundenframes und als solche nicht in der Liste mit abgespeichert. Wenn also bei einem gestoppten Auftrag der Auftrag geändert wird, ist der Benutzer der Zentrale selbst für die Nebenwirkungen verantwortlich.
	Sekundenframe	Ältester Sekundenframe (Achtung: Dieser Frame enthält eine Liste aus Auftragsframes!)
101	**GetYoungest**	Jüngstes Listenelement lesen
	Ausgabeparameter	
	RetCode : RetCode	OK: jüngster Sekundenframe richtig geliefert O_SF: Liste enthält überhaupt keine Sekundenframes
	PosNr : ULONG	Positionsnummer des gelieferten SF
	Listenversion: ui2	Versionskennung, die bei jeder Auftragsänderung vom Feldgerät geändert wird. Die Versionsnummer wird auch bei beim Start der Liste zurückgegeben, so dass die Zentrale feststellen kann, ob eine Änderung stattgefunden hat. Wie unmittelbar aus den Datenstrukturen ersichtlich ist, ist die Versionsnummer der Liste nicht die Versionsnummer des Sekundenframes und als solche nicht in der Liste mit abgespeichert. Wenn also bei einem gestoppten Auftrag der Auftrag geändert wird, ist der Benutzer der Zentrale selbst für die Nebenwirkungen verantwortlich.
	Sekundenframe	Jüngster Sekundenframe (Achtung: Dieser Frame enthält eine Liste aus Auftragsframes!)

Liste		
METHOD	**Name**	**Beschreibung**
102	**GetSFSince**	Sekundenframes ab übergebener Zeit/Positionsnummer in der Reihenfolge ihres Entstehens lesen. Es werden nur Sekundenframes zurückgegeben, die im Ringpuffer nach diesem Element eingetragen wurden (Ohne Zeitumstellung sind das nur jüngere Elemente. Bei Zeitumstellungen wird in jedem Fall nach der Reihenfolge des Eintragens ausgegeben, nicht nach dem Zeitstempel) Existiert kein Sekundenframe zu Zeitstempel/Positionsnummer, so beginnt diese Methode mit dem als ersten in den Ringpuffer eingetragene Element, dessen UTC-Zeitstempel ‚jünger' ist als die übergebene Zeit (und geht anschließend in der Reihenfolge des Eintrags weiter). Existiert auch dieses nicht, liefert die Methode RetCode==NO_SF, bei erfolgreichem Aufruf wird RetCode== SF_FOLLOW oder RetCode==SF_NOFOLLOW geliefert. Diese Methode verändert weder den aktuellen Füllgrad noch den Füllgrad bei welchem die Liste auslöst.
	Eingabeparameter	
	Zeit : ZEITSTEMPEL_UTC	Zeitpunkt ab welchem Elemente gelesen werden
	PosNr : ui4	Positionsnummer ab der gelesen wird. Das erste zu liefernde Element ist das auf Zeit.PosNr folgende Element. Nullvalue=0xffffffff Die Positionsnummer 0xffffffff darf nicht auftreten.
	MaxAnzahl	Maximale Anzahl zu lesender Elemente
	Ausgabeparameter	
	RetCode : RetCode	SF_FOLLOW: Sekundenframes richtig geliefert und weitere Sekundenframes, die später als diese eingetragen wurden, in der Liste SF_NOFOLLOW: Sekundenframes richtig geliefert und keine später eingetragenen Sekundenframes im Archiv. NO_SF: Liste enthält keinen Sekundenframe, der die Bedingung erfüllt.
	AbZeit : ZEITSTEMPEL_UTC	Zeitstempel des Sekundenframes, der direkt vor den übertragenen Sekundenframes im Ringpuffer eingetragen ist oder 0 falls kein solcher Sekundenframe im Ringpuffer ist.
	AbPosNr : ULONG	Positionsnummer des Sekundenframes, der direkt vor den übertragenen Sekundenframes im Ringpuffer eingetragen ist oder undefiniert falls kein solcher Sekundenframe im Ringpuffer ist. Wenn nacheinander mehrfach mit GetSFSince ausgelesen wird und *keine* Elemente überschrieben wurden, ist ‚AbZeit dieses Aufrufs'== ‚Zeit des letzten Sekundenframes des letzten Aufrufs' und ‚AbPosNr dieses Aufrufs' == ‚PosNr des letzten Sekundenframes des letzten Aufrufs'. Existiert zum ersten gesendeten kein älteres Element mehr, ist AbPosNr undefiniert und AbZeit == 0.

Liste		
METHOD	**Name**	**Beschreibung**
	BisZeit: ZEITSTEMPEL_UTC	Zeitstempel des letzten im folgenden gesendeten Elements also von Element[Anzahl-1].
	BisPosNr : ui4	Positionsnummer des letzten im folgenden gesendeten Elements also von Element[Anzahl-1].
	Listenversion: ui2	Versionskennung, die bei jeder Auftragsänderung vom Feldgerät geändert wird. Die Versionsnummer wird auch bei Auftragsänderungen zurückgegeben, so dass die Zentrale feststellen kann, ob eine Änderung stattgefunden hat. Wie unmittelbar aus den Datenstrukturen ersichtlich ist, ist die Versionsnummer der Liste nicht die Versionsnummer des Sekundenframes und als solche nicht in der Liste mit abgespeichert. Wenn also bei einem gestoppten Auftrag der Auftrag geändert wird, ist der Benutzer der Zentrale selbst für die Nebenwirkungen verantwortlich.
	Anzahl : ui2	Anzahl der folgenden Sekundenframes
	Sekundenframes [] : Sekundenframe	Gelesene Meldungen (Achtung: Jede Meldung enthält wieder eine Liste aus Meldungsteilen)
103	**GetSFSinceWithEvent**	GetSFSinceWithEvent ist eine Kombination der Methoden GetSFSince() und SetEvent(). Die Parameter LastTime und LastPosNr von SetEvent sind die des letzten von GetSFSince zurückgegebenen Sekundenframes. Diese Methode darf nur von dem als EventDestination eingetragenen Gerät (Zentrale) aus aufgerufen werden. Hinweis: Dieser Befehl ist nicht für Systemzugänge gedacht. Die Zentrale muss solche Befehle abfangen und negativ quittieren (ACCESS_DENIED), wenn sie über die Zentrale geroutet werden. Befehle, die direkt an das Feldgerät gesendet werden, werden aufgrund der Quell-IP-Adresse erkannt und dort abgefangen.
	Eingabeparameter	
	Zeit : utc	Zeitpunkt ab welchem Elemente gelesen werden
	PosNr : ui4	Positionsnummer ab der gelesen wird. Das erste zu liefernde Element ist das auf Zeit.PosNr folgende Element.
	MaxAnzahl	Maximale Anzahl zu lesender Elemente
	Fill: UBYTE	Füllgrad in %, bei dem der Event ausgelöst wird. Siehe SetEvent.
	AuthenticateAnswer	0 (FALSE): Der Respond braucht nicht authentifiziert zu werden (keine SHA-1 Prüfsumme notwendig). Wenn trotzdem authentifiziert wird, muss die Prüfsumme stimmen. 1 (TRUE): Der Respond muss authentifiziert werden.
	Ausgabeparameter	

Liste		
METHOD	**Name**	**Beschreibung**
	RetCode : ui2	SF_FOLLOW: Sekundenframes richtig geliefert und weitere Sekundenframes, die später als diese eingetragen wurden, in der Liste enthalten. SF_NOFOLLOW: Sekundenframes richtig geliefert und keine später eingetragenen Sekundenframes im Archiv. NO_EVENT, wenn aus irgendeinem Grund der Event nicht eingetragen werden kann. NO_SF: Liste enthält keinen Sekundenframe, der die Bedingung erfüllt. ACCESS_DENIED Zugriff nicht erlaubt, da nicht von Event-Destination ausgelöst.
	AbZeit : utc	Zeitstempel des Sekundenframes, der direkt vor den übertragenen Sekundenframes im Ringpuffer eingetragen ist oder 0 falls kein solcher Sekundenframe im Ringpuffer ist.
	AbPosNr : ui4	Positionsnummer des Sekundenframes, der direkt vor den übertragenen Sekundenframes im Ringpuffer eingetragen ist oder undefiniert falls kein solcher Sekundenframe im Ringpuffer ist. Wenn nacheinander mehrfach mit GetSFSince ausgelesen wird und *keine* Elemente überschrieben wurden, ist ‚AbZeit dieses Aufrufs'== ‚Zeit des letzten Sekundenframes des letzten Aufrufs' und ‚AbPosNr dieses Aufrufs' == ‚PosNr des letzten Sekundenframes des letzten Aufrufs'. Existiert zum ersten gesendeten kein älteres Element mehr, ist AbPosNr undefiniert und AbZeit == 0.
	BisZeit: ZEITSTEMPEL_UTC	Zeitstempel des letzten im folgenden gesendeten Elements also von Element[Anzahl-1].
	BisPosNr : ui4	Positionsnummer des letzten im folgenden gesendeten Elements also von Element[Anzahl-1].
	Listenversion: ui2	Versionskennung, die bei jeder Auftragsänderung vom Feldgerät geändert wird. Die Versionsnummer wird auch bei Auftragsänderungen zurückgegeben, so dass die Zentrale feststellen kann, ob eine Änderung stattgefunden hat. Wie unmittelbar aus den Datenstrukturen ersichtlich ist, ist die Versionsnummer der Liste nicht die Versionsnummer des Sekundenframes und als solche nicht in der Liste mit abgespeichert. Wenn also bei einem gestoppten Auftrag der Auftrag geändert wird, ist der Benutzer der Zentrale selbst für die Nebenwirkungen verantwortlich.
	Anzahl : ui2	Anzahl der folgenden Sekundenframes
	Sekundenframes [] : Sekundenframe	Gelesene Meldungen (Achtung: Jede Meldung enthält wieder eine Liste aus Meldungsteilen)

Liste		
METHOD	**Name**	**Beschreibung**
104	**SetEvent**	Beauftragt die Liste beim überschreiten des angegebenen Füllgrads „Fill“ die Methode EvList::OnFull() im durch SetEventDestination() angegebenen Gerät (in der Regel die Zentrale) aufzurufen. Die Übergabeparameter LastTime und LastPosNr markieren die Startposition zur Ermittelung des aktuellen Füllgrads. Jedes mal wenn nach dem Eintragen eines Sekundenframes der Füllgrad überschritten wird, wird die Methode OnFull (0, ArchivEvent-Objekt) aufgerufen. Wenn beim Füllgrad ein Wert > 100 eingetragen wird, wird kein neues Event ausgelöst. Bei einem Füllgrad von 0 wird das Event nach jedem neuen Eintrag ausgelöst.
	Eingabeparameter	
	LastTime	Zeitpunkt des letzten Elements, das abgeholt wurde
	LastPosNr	Positionsnummer des letzten Elements, das abgeholt wurde
	Fill	Maximaler Füllgrad, bei dem EvList::OnFull ausgelöst wird. Fill=0 → nach jedem Eintrag OnFull auslösen Fill>100 → nie OnFull auslösen
	Ausgabeparameter	
	RetCode : ui2	OK, wenn das Event eingetragen werden kann; NO_EVENT, wenn aus irgendeinem Grund das Event nicht eingetragen werden kann ACCESS_DENIED Zugriff nicht erlaubt, da nicht von EventDestination ausgelöst.
105	**Start**	Startet die Liste, d.h. die Aufträge werden scharf. Auch die Aggregation von AEAggregiert startet. Wenn die Liste gestartet wird, wird der Ringpuffer gelöscht. Es ist bei bestimmten Listen herstellerabhängig möglich, dass sie nicht gestoppt (und gestartet) werden können.
	Eingabeparameter	
		Keiner
	Ausgabeparameter	
	RetCode	OCIT_OK : Datenerfassung ist gestartet NOT_POSSIBLE: Liste kann prinzipiell nicht gestartet werden. NOT_INACTIVE: Datenerfassung ist bereits gestartet (Liste wurde nicht gelöscht) ERROR: Befehl nicht ausführbar (Auftraege unvollständig konfiguriert)

Liste		
METHOD	**Name**	**Beschreibung**
	Listenversion	Versionsnummer, die auch bei GetSFSinceXXX zurückgegeben wird.
106	**Stop**	Hält die Datenerfassung der Liste an. Es ist nach dem Stop der Liste noch möglich, den Ringpuffer abzuholen. Es ist bei bestimmten Listen herstellerabhängig möglich, dass sie nicht gestoppt (und gestartet) werden können.
	Eingabeparameter	
		Keiner
	Ausgabeparameter	
	RetCode	OCIT_OK : Liste ist gestoppt, wird auch bei einer bereits gestoppten Liste zurückgegeben. NOT_POSSIBLE : Liste kann prinzipiell nicht gestoppt werden. ERROR : Befehl konnte nicht ausgeführt werden.
107	**Reset**	Hält die Liste an. Entfernt alle mit AddAuftrag angefügten Aufträge wieder aus der Liste. Setzt die EventDestination auf die Zentrale. Wenn eine EventDestination eingetragen war, wird an die alte EventDestination ein OnInvalidate gesendet.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OCIT_OK RetCode != OCIT_OK -> Befehl konnte nicht ausgeführt werden.
	ListenversionAlt, ListenversionNeu	Listenversion vor Reset Listenversion nach Reset
108	**AddAuftrag**	Fügt einen neuen Auftrag zu der Liste hinzu. OCIT-Outstations schreibt nur vor, dass neue Aufträge in einer gestoppten Liste eingefügt werden können. Es bleibt dem Hersteller unbenommen, dies auch für bereits gestartete Listen zu implementieren. Die neue Listenversion kommt erst beim Start des neu hinzugefügten Auftrags hinzu. Der neue Auftrag ist deaktiviert, solange die Liste deaktiviert ist und wird beim Start der Liste dann ebenfalls gestartet. Bei einer aktivierten Liste ist der neue Auftrag ebenfalls deaktiviert, bis er durch einen Start-Befehl beim Auftrag aktiviert wird.
	Eingabeparameter	
	Member : USHORT OType : USHORT	Gibt den Typ des anzufügenden Auftrags an. Die angegebene Domain muss vom Auftrag abgeleitet sein.
	Ausgabeparameter	

Liste		
METHOD	**Name**	**Beschreibung**
	RetCode	OCIT_OK PARAM_INVALID mit Member/OType angegebener Auftragstyp ist nicht bekannt, Auftrag wurde nicht angelegt. NOT_INACTIVE der Auftrag darf nicht gestartet sein um diese Methode auszuführen BUFFER_TOO_SMALL: wird geliefert, wenn der Sekundenframe so groß werden kann, dass weniger als vier Einträge in den Ringpuffer passen. NOT_POSSIBLE: Anzahl Aufträge zu groß (max. 255 Aufträge möglich)
	AuftragsNr : UBYTE	Auftragsnummer des neu angefügten Auftrags falls RetCode ==OK. Der neue Auftrag kann mit dem Pfad Liste()/AuftragsNr() angesprochen werden.
	ListenversionAlt, ListenversionNeu	Listenversion vor AddAuftrag Listenversion nach AddAuftrag
109	**SetEventDestination**	Setzt das Ziel aller Events dieser Liste. Wenn ein Event-Ziel gesetzt wird, löst diese Methode den Event OnInvalidate beim alten Event-Ziel aus. Achtung: Zuerst wird die EventDestination gesetzt und dann der Event aufgerufen. Die EventDestination wird auch dann nicht zurückgenommen, wenn der Event OnInvalidate mit Timeout-Fehler zurückkehrt! Gesetzte Events bleiben gesetzt.
	Eingabeparameter	
	ZNr, FNr : USHORT	ZNr/FNr des Gerätes, welches die Traps empfangen soll.
	Ausgabeparameter	
	RetCode	RetCode != OCIT_OK -> Befehl konnte nicht ausgeführt werden
	ListenversionAlt, ListenversionNeu	Listenversion vor SetEventDestination Listenversion nach SetEventDestination
110	**SetSize**	Setzt die Größe des Ringpuffers in Byte. Das Gerät entscheidet, wie groß die Liste wirklich sein kann und liefert den eingestellten Wert zurück. Die Größe muss gesetzt werden, während die Liste offline ist und löscht alle noch vorhandenen Einträge in der Liste. Das Gerät versucht die gewünschte Größe des Puffers in etwa zu treffen.
	Eingabeparameter	

Liste		
METHOD	**Name**	**Beschreibung**
	Persistenz : UBYTE	Gibt an welche Teile der Liste nach Netzausfall erhalten bleiben: Keine, die gesamte Liste ist nach Netzausfall auf Standardwerte zurückgesetzt. Aufträge, die Listenaufträge bleiben erhalten, die Daten (Inhalt des Ringpuffers) gehen verloren. Nach Netzausfall wird die Liste automatisch wieder gefüllt. Alles, sowohl die Listenaufträge als auch der Inhalt des Ringpuffers bleiben über einen Netzausfall erhalten.
	ListeSizeP : UBYTE	Minimale gewünschte Größe der Liste in Prozent des restlich verfügbaren Speichers. Es wird die Größe genommen, die zu mehr Bytes führt, aber nicht mehr als 100%. Die anderen Listen werden nicht verkleinert.
	Ausgabeparameter	
	RetCode	OCIT_OK: Der Befehl wurde ausgeführt NOT_INACTIVE: Die Liste darf nicht gestartet sein.
	CurrentPersistenz : UBYTE	eingestellte Persistenz dieser Liste.
	CurrentSizeB : ULONG	Größe der Liste in Byte, die eingestellt wurde, bzw. Größe der aktuellen Liste, falls keine Änderung möglich ist.
	CurrentSizeP : UBYTE	Größe der Liste in Prozent des vor dem Aufruf verfügbaren Speichers.
	ListenversionAlt, ListenversionNeu	Listenversion vor SetSize Listenversion nach SetSize
111	**SetOverwriteOnFull**	Legt das Verhalten bei vollem Ringpuffer fest.
	Eingabeparameter	
	OverwriteOnFull : BOOL	Falls gesetzt (true), überschreibt die Liste die alten Datenframes (Ringpuffer). Andernfalls stoppt die Liste. Die Defaulteinstellung nach Reset ist true (Ringpuffer).
	Ausgabeparameter	
	RetCode	OK, NOT_POSSIBLE falls das Gerät diese Einstellung nicht unterstützt.
112	**Suspend**	Hält die Erfassung der Liste an, ohne sie richtig zu stoppen. Der Ringpuffer bleibt erhalten. Wenn die Liste suspendiert wurde, wird die Betriebsmeldung Suspend in der Liste gespeichert.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
		OCIT_OK: Der Befehl wurde ausgeführt.

Liste		
METHOD	**Name**	**Beschreibung**
113	**Unsuspend**	Wenn die Liste mit Supend angehalten ist, wird dieses „Suspend“ wieder zurückgenommen und die Betriebsmeldung Unsuspend in den Ringpuffer der Liste gespeichert.

4.2.4.3 Eventhandler für Listen in der Zentrale

Für alle Listen aller Feldgeräte existiert in der Zentrale **eine** Instanz des OBJTYPE EvList. Ein Event ist gleich, wenn Ziel (Zentrale), Methode und Eingabeparameter übereinstimmen.

Das Feldgerät sendet das nächste Event erst nachdem es eine Quittung auf das vorangegangene gleiche Event erhalten hat oder wenn es erneut aktiviert wurde (mit Liste::SetEvent(), Liste::GetSFSinceWithEvent(), Auftrag::ActivateEvent()). Dies ermöglicht eine Flusskontrolle für Events unter Berücksichtigung der Übertragungsdauer der Übertragungsstrecke sowie der Verarbeitung der Events durch die Zentrale. Die Übertragungsgeschwindigkeit der Events passt sich an die vorhandene Bandbreite und das aktuelle Übertragungsvolumen an.

Die Eventhandler werden folgendermaßen aufgerufen:

EvList		
METHOD	**Name**	**Beschreibung**
200	**OnFull**	Wird vom Feldgerät in der EventDestination (meist gleich Zentrale) aufgerufen, wenn der Füllgrad überschritten ist.
	Eingabeparameter	
	ZNr : USHORT FNr : USHORT	Absender Feldgerät
	Liste : UBYTE	Listennummer der Liste deren Füllstand überschritten wurde.
	Ausgabeparameter	
	RetCode	Wird ignoriert. Nur notwendig, um Sendebestätigung zur Flußkontrolle zu erhalten. Die Anwenderfunktion sollte immer OK zurückgeben.
201	**OnInvalidate**	Wird vom Feldgerät in der EventDestination (meist gleich Zentrale) aufgerufen, wenn ein anderes Event-Ziel gesetzt wird. Siehe Liste::SetEventDestination().
	Eingabeparameter	
	ZNr:USHORT, FNr:USHORT	Auslösendes Feldgerät
	ListenNr : UBYTE	Liste deren Event neu gesetzt wurde.
	ZNrNeu:USHORT, FNrNeu:USHORT	Gerätenummer, des Gerätes welches das neue Event beantragt hat.

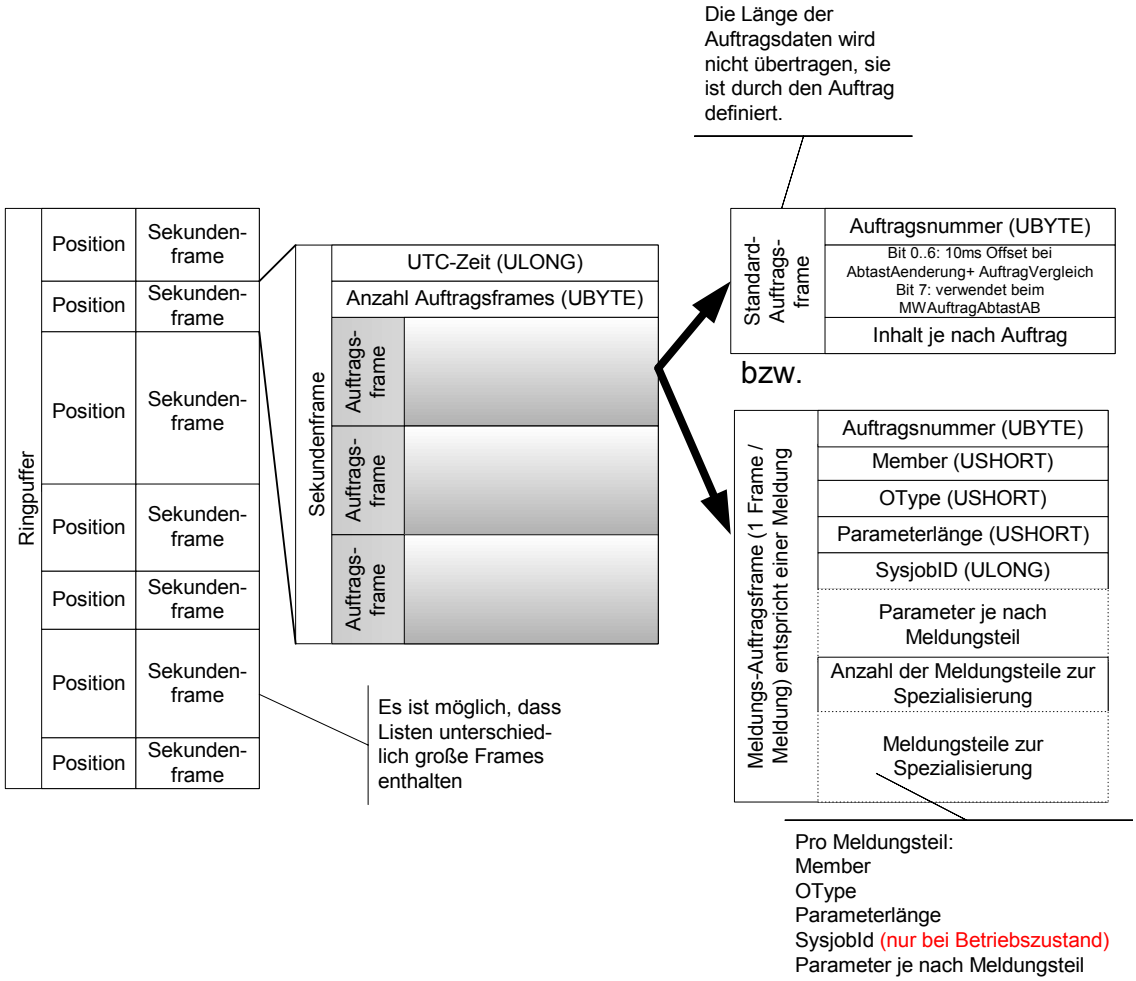
EvList		
METHOD	**Name**	**Beschreibung**
	Ausgabeparameter	
	RetCode	Wird ignoriert. Nur notwendig, um Sendebestätigung zur Flußkontrolle zu erhalten. Die Anwenderfunktion sollte immer OK zurückgeben.
202	**OnInsert**	Wird vom Feldgerät in der EventDestination (meist gleich Zentrale) aufgerufen, wenn die dynamischen Daten eines ‚aktivierten' Auftrags eingetragen wurden. Siehe z.B. ActivateEvent
	Eingabeparameter	
	ZNr:USHORT, FNr:USHORT	Auslösendes Feldgerät
	ListenNr : UBYTE	Liste deren Event neu gesetzt wurde.
	AuftragNr : UBYTE	Auftrag, der das Event ausgelöst hat.
	Ausgabeparameter	
	RetCode	Wird ignoriert. Nur notwendig, um Sendebestätigung zur Flußkontrolle zu erhalten. Die Anwenderfunktion sollte immer OK zurückgeben.
203	**OnNetzAus**	wird vom Feldgeraet (meist gleich Zentrale) aufgerufen, wenn ein Netzausfall erkannt wird
	Eingabeparameter	
	ZNr:USHORT, FNr:USHORT	Auslösendes Feldgerät
	Vorgangsken- nung:SYSJOBID	Vorgangskennung der Störung, identisch mit Vorgangskennung der korrespondierenden NetzAus-Meldung im Standard-Meldearchiv
	Ausgabeparameter	
	RetCode	Wird ignoriert. Nur notwendig, um Sendebestätigung zu erhalten. Die Anwenderfunktion sollte immer OK zurückgeben.

4.2.5 Sekundenframe / Auftragsframe

Jeder Sekundenframe besteht aus einem oder mehreren Auftragsframes. Ein Auftragsframe besteht aus einer Auftragsnummer (1 Byte) sowie den dynamischen Daten, die der Auftrag generiert hat. Diese Daten beinhalten bei Messwerten im Normalfall nur die reinen Sachdaten, bei Meldungen wird zusätzlich die Member/OType-Kennung gesendet, die die ‚Meldungsnummer' des Meldungsteils repräsentiert. Zu jedem Meldungsteil kann der Hersteller in der TYPE-Datei optional einen Formatstring angeben, mit dessen Hilfe die Zentrale den Mel-

dungsteil in ein lesbares Format übersetzt. Fehlt der Formatstring, gibt die Zentrale die Daten in einem beliebigen Format aus.

Bei Meldungen folgt auf die Member/OType-Kennung der Parametersatz, der die Meldung genauer beschreibt. Der Aufbau und die Länge der Parametersätze ist von der Member/OType-Kennung abhängig. Um unbekannte Meldungsteile überspringen zu können, wird bei jedem Meldungsteil nach dem Member/OType-Feld eine 2Byte Länge des Parametersatzes gespeichert.



Schema Sekunden- und Auftragsframe

4.2.6 Auftrag

Zu jeder Liste gehört eine Reihe von Aufträgen die festlegen, welche dynamischen Daten in der Liste abgespeichert werden. Jeder Auftrag gehört zu genau einer Liste, listenübergreifende Aufträge sind nicht vorgesehen.

Innerhalb einer Liste ist ein Auftrag eindeutig durch ein UBYTE (Auftragsnummer) identifiziert. Der Auftrag ist damit durch den Pfad `Geraet/Liste()/Auftrag()` zu erreichen. Bei dynamischen Listen ist ein Auftrag nicht von vornherein vorhanden, sondern muss vorher

über die Methode „Liste.AddAuftrag“ angelegt werden. Bei statischen Listen (z.B. Meldungen) sind die Aufträge bereits vorhanden.

Anders als die Liste ist der Auftrag eine „virtuelle Basisklasse“, d.h. es gibt eine Reihe von speziellen Aufträgen, die in Listen verwendet werden können, aber nicht der Basisauftrag selbst. Welcher Auftragstyp verwendet werden soll, wird in „Liste.AddAuftrag“ angegeben. Es ist möglich, dass nicht jede Liste jeden Auftragstyp akzeptiert.

Bestimmte Auftragstypen können sich aus Auftragselementen zusammensetzen, in denen dann gespeichert wird, welche Daten übertragen werden sollen. Andere Auftragstypen enthalten bereits implizit die Daten, die dort übertragen werden, z.B. der Auftragstyp für Meldungen.

- Es gibt unterschiedliche Typen von Aufträgen
- Alle Aufträge sind ObjTypes unter der jeweiligen Liste (Pfad: Liste()/Auftrag())
- Ein Auftrag aggregiert mehrere Auftragselemente.

Jeder Auftrag enthält implizit eine Bedingung. Wenn die Bedingung erfüllt ist, werden die dynamischen Daten des Auftrags erfasst und in einem Sekundenframe des Ringpuffers gespeichert.

Jeder Auftrag kann ggf. gestartet und gestoppt werden. Das Starten und Stoppen eines Auftrags bedeutet nur, dass die Werte für diesen Auftrag nicht erfasst und nicht in den Ringpuffer eingetragen werden. Die Liste bleibt mit den anderen Aufträgen weiterhin in Betrieb. In gestopptem Zustand kann der Auftrag ggf. geändert werden. Um die Liste sinnvoll auswerten zu können, werden Start und Stop eines Auftrages sowie das Anhalten der gesamten Liste und Zeitsprünge als Meldungen in der Liste eingetragen. Wenn der Auftrag geändert und dann wieder aktiviert wird, wird entsprechend die Versionsnummer der Liste geändert und in den späteren GetSFSinceXXX-Antworten ebenfalls die geänderte Listennummer zurückgegeben.

Es ist möglich, dass Aufträge existieren, die nicht gestartet und gestoppt werden können. Dies gilt insbesondere für die Default-Meldungslisten und ihre Aufträge. Ebenfalls ist es möglich, dass in bestimmten Listen keine neuen Aufträge hinzugefügt werden können. Alle neu angelegten Aufträge sind deaktiviert. Ein Auftrag kann nicht einzeln aktiviert werden, wenn die Liste deaktiviert ist. Beim Start der Liste werden alle Aufträge (unabhängig davon, ob sie zwischenzeitlich aktiviert wurden oder nicht) aktiviert. Wenn die Liste läuft, kann der Auftrag einzeln gestoppt und gestartet werden.

Beim Start einer Liste und beim Start eines Auftrags während des Laufes einer Liste wird immer eine (neue) Listenversion zurückgegeben.

Um sicherzustellen, dass keine Terminals die Aufträge von Listen ändern können, gilt: Das Feldgerät weist alle Änderungsaufträge zurück, die nicht von der EventDestination stammen.

4.2.6.1 Methoden aller Aufträge

Auftrag		
METHOD	**Name**	**Beschreibung**

Auftrag		
METHOD	**Name**	**Beschreibung**
120	**AddElement**	Allgemeine Funktion für alle Aufträge. Wird aufgerufen, wenn ein neues Element zum Auftrag hinzuzufügen ist. AddElement kann nur aufgerufen werden, wenn der Auftrag noch nie aktiviert worden ist. Dadurch wird erreicht, dass die Abfrage der Auftragselemente auch für „alte“ Daten möglich ist.
	Eingabeparameter	
	Member : USHORT OType : USHORT	Auftragselement, welches hinzuzufügen ist
	Ausgabeparameter	
	RetCode	OK : wird zurückgeliefert, wenn das Auftragselement hinzugefügt werden konnte NOT_POSSIBLE : wird aufgerufen, wenn der Auftragstyp keine Auftragselemente zulässt, wie z.B. bei Meldungen, R09 und AMLi NOT_INACTIVE: wird zurückgegeben, wenn zwar theoretisch Auftragselemente hinzugefügt werden könnten, aber der Auftrag aktiv ist.
	AENr : UBYTE	Nummer des Auftragselements, das hinzugefügt wurde. Bei != OK ist der Wert undefiniert.
	ListenversionAlt, ListenversionNeu	Listenversion vor AddElement Listenversion nach AddElement
121	**Start**	startet den Auftrag, wenn die Liste aktiv ist (und es möglich ist, den Auftrag getrennt zu starten). Wenn die Liste nicht aktiv ist, wird ein Fehler zurückgegeben. Start löst einen Meldungseintrag StartAuftrag aus, der ggf. in die Liste selbst eingetragen wird. Diese Methode wird in OCIT-Outstations Version 1 nicht realisiert.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Auftrag erfolgreich aktiviert wurde. NOT_POSSIBLE : wird geliefert, wenn der Auftragstyp (bzw. die Liste) nicht zulässt, dass der Auftrag gestartet werden kann. NOT_ACTIVE : wird geliefert, wenn die Liste inaktiv ist.
	Listenversion : USHORT	Neue Versionsnummer der Liste, zu der der Auftrag gehört. Wenn der RetCode != OK ist, ist der Wert undefiniert.

Auftrag		
METHOD	Name	Beschreibung
122	**Stop**	Stoppt den Auftrag, wenn die Liste aktiv ist (und es möglich ist, den Auftrag getrennt zu stoppen). Wenn die Liste nicht aktiv ist, wird ein Fehler zurückgegeben. Stop löst einen Meldungseintrag StopAuftrag aus, der ggf. in die Liste selbst eingetragen wird.. Diese Methode wird in OCIT-Outstations Version 1 nicht realisiert.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Auftrag erfolgreich aktiviert wurde. NOT_POSSIBLE : wird geliefert, wenn der Auftragstyp (bzw. die Liste) nicht zulässt, dass der Auftrag gestoppt werden kann. NOT_ACTIVE : wird geliefert, wenn die Liste inaktiv ist.

4.2.6.2 Meldungsauftrag

Bei jeder Meldungsliste gibt es pro MeldungsDegree (Information, Warnung, Fehler, schwerer Fehler) einen Meldungsauftrag. Es ist nicht notwendig, dass in einer Meldungsliste alle MeldungsDegree (und damit Meldungsaufträge) vorhanden sein müssen. Welche „Meldungen“ von welchem Auftrag behandelt werden, wird durch den Degree des Haupt-Meldungsteils festgelegt. Ein Meldungsauftrag trägt nur die Meldungen in seine Liste ein, bei denen der Degree des Hauptmeldungsteils gleich dem eigenen Degree ist.

Bei Meldungsaufträgen, die standardmäßig Bestandteil von Archiven sind, ist die Einordnung von Meldungen in die Include- bzw. Exclude-Liste archivspezifisch vorgegeben.

Bei Meldungsaufträgen, die eine Zentrale anlegt, gelten folgende Festlegungen:

- Ein Meldungsauftrag besitzt nach dem Anlegen durch die Zentrale den Degree I
- Alle Meldungen befinden sich in der Exclude-Liste, d.h. standardmäßig generiert der Auftrag keine Meldungsframes. Nach ResetMT werden alle Meldungen, die dem Degree des Auftrags entsprechen, in die Include-Liste aufgenommen.

Die Elemente können über die Get-Funktion der jeweiligen Auftragselemente abgefragt werden. Auf ein detailliertes Lock und Unlock wird verzichtet, da eine Liste nur von einem Teilnehmer „verwaltet“ wird.

Interface der Meldungsaufträge

Meldungsauftrag

METHOD	Name	Beschreibung
120, 121, 122	**AddElement, Start, Stop**	Siehe 4.2.6
130	**IncludeMT**	Wird aufgerufen, wenn ein Haupt-Meldungsteil diesem MeldungsDegree (und damit dem Meldungsauftrag) zugeschlagen werden soll. Es ist damit möglich, dass pro Haupt-Meldungsteil mehr als eine Meldung (mit unterschiedlichen MeldungsDegree) entsteht. Pro Meldungsauftrag wird natürlich nur eine Meldung übertragen. Wenn der Eintrag in der Exclude-Liste bereits vorhanden ist, wird er aus der Exclude-Liste entfernt.
	Eingabeparameter	
	Member : USHORT OType : USHORT	Haupt-Meldungsteil
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Meldungsteil erfolgreich eingetragen wurde. NOT_POSSIBLE : wird geliefert, wenn in dieser Liste Meldungsteile nicht ein- oder ausgehängt werden können. PARAM_INVALID : der angegebene Meldungsteil ist kein dem Gerät bekannter Haupt-Meldungsteil (oder ein ganz anderes Objekt). NOT_INACTIVE : Obwohl die Änderungen während des Laufes unproblematisch sind, kann herstellerabhängig (und ggf. projektabhängig) gefordert werden, dass in einer aktiven Liste keine Änderungen im In- und Exclude gemacht werden. In diesem Fall wird NOT_INACTIVE zurückgegeben.
	ListenversionAlt, ListenversionNeu	Listenversion vor IncludeMT Listenversion nach IncludeMT
131	**ExcludeMT**	Wird aufgerufen, wenn ein Haupt-Meldungsteil aus diesem Meldungsdegree (und damit dem Meldungsauftrag) entfernt werden soll. Es ist damit möglich, dass ein Haupt-Meldungsteil nicht mehr übertragen wird. Wenn der Eintrag in der Include-Liste bereits vorhanden ist, wird er aus der Include-Liste entfernt.
	Eingabeparameter	
	Member : USHORT OType : USHORT	Haupt-Meldungsteil
	Ausgabeparameter	

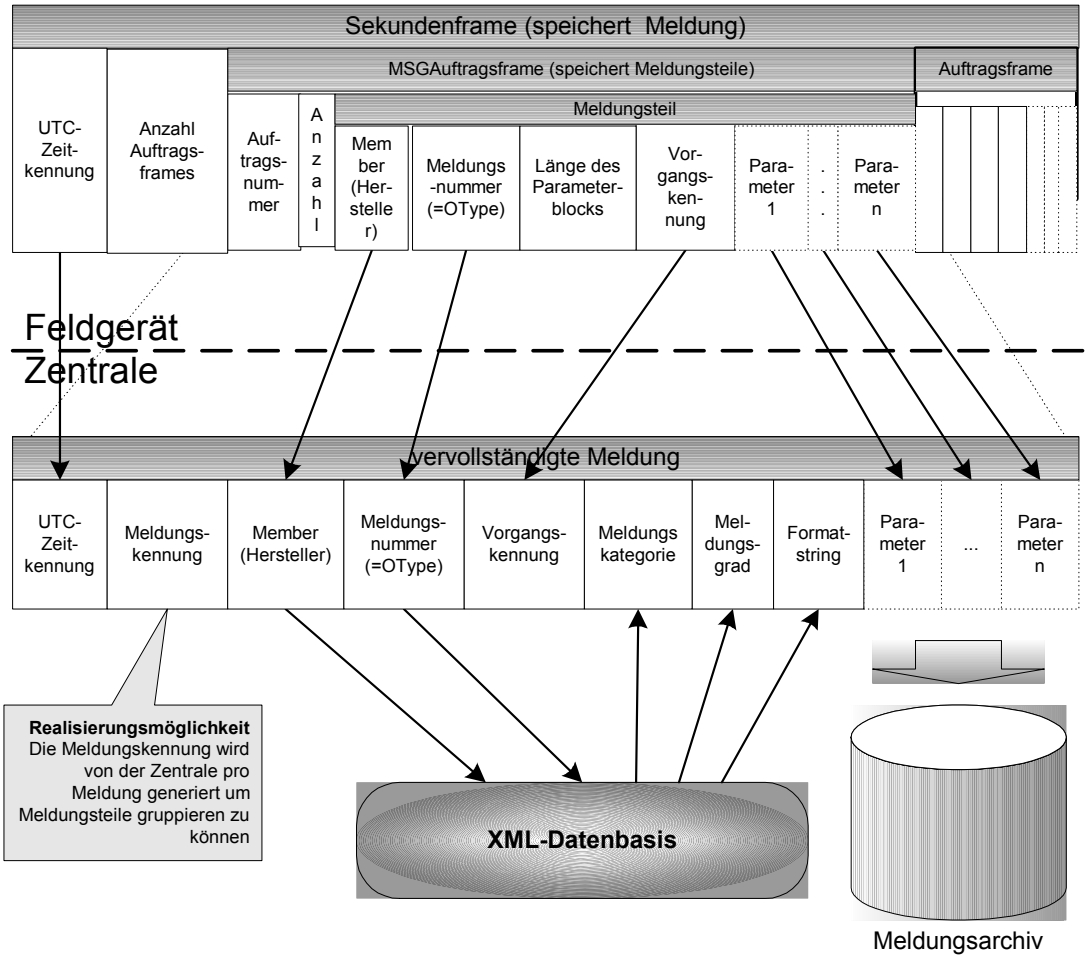
Meldungsauftrag		
METHOD	**Name**	**Beschreibung**
	RetCode	OK: wird zurückgeliefert, wenn der Meldungsteil erfolgreich eingetragen wurde. NOT_POSSIBLE : wird geliefert, wenn in dieser Liste Meldungsteile nicht ein- oder ausgehängt werden können. PARAM_INVALID : der angegebene Meldungsteil ist kein dem Gerät bekannter Haupt-Meldungsteil (oder ein ganz anderes Objekt). NOT_INACTIVE : Obwohl die Änderungen während des Laufes unproblematisch sind, kann herstellerabhängig (und ggf. projektabhängig) gefordert werden, dass in einer aktiven Liste keine Änderungen im In- und Exclude gemacht werden. In diesem Fall wird NOT_INACTIVE zurückgegeben.
	ListenversionAlt, ListenversionNeu	Listenversion vor ExcludeMT Listenversion nach ExcludeMT
132	**ResetMT**	Die Include und Exclude-Liste wird auf geräteinterne Defaultwerte zurückgesetzt.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	RetCode	OK: wird zurückgeliefert, wenn der Meldungsteil erfolgreich eingetragen wurde. NOT_POSSIBLE : wird geliefert, wenn in dieser Liste Meldungsteile nicht ein- oder ausgehängt werden können.
	ListenversionAlt, ListenversionNeu	Listenversion vor ResetMT Listenversion nach ResetMT
133	**GetInEx**	Die Include und Exclude-Liste wird zurückgegeben. Alle vom Gerät generierten Meldungen befinden sich entweder in der Include- oder Exclude-Liste des Auftrags.
	Eingabeparameter	
		Keine
	Ausgabeparameter	
	IncludeAnzahl : USHORT IncludeMT[] : {Member, OType}	Liste der aktuellen Include Meldungsteile.
	ExcludeAnzahl : USHORT ExcludeMT[] : {Member, OType}	Liste der aktuellen Exclude Meldungsteile.
	RetCode	OK
119	**ActivateEvent**	Aktiviert bzw. deaktiviert den Event, der ausgelöst wird, wenn in die Liste ein Element dieses Auftrags kommt.
	Eingabeparameter	

Meldungsauftrag		
METHOD	**Name**	**Beschreibung**
	ActivateIt : UBYTE	0: Der Event wird deaktiviert 1: Der Event wird aktiviert
	Ausgabeparameter	
	RetCode	OK: Der Event wurde erfolgreich aktiviert bzw. deaktiviert.
	ListenversionAlt, ListenversionNeu	Listenversion vor ActivateEvent Listenversion nach ActivateEvent
134	**SetDegree**	Versieht den Meldungsauftrag mit einem neuen Meldungs-degree. Die Include-/Exclude-Liste wird nicht verändert.
	Eingabeparameter	
	Meldungsdegree : UByte	Neuer Meldungsdegree
	Ausgabeparameter	
	RetCode	OK: Der Meldungsdegree wurde erfolgreich zugewiesen
	ListenversionAlt, ListenversionNeu	Listenversion vor SetDegree Listenversion nach SetDegree

4.2.7 Meldung

Im gemeinsamen Meldungs- und Messwertmodell gibt es keine Meldung an sich. Stattdessen wird pro Meldung ein MSGAuftragFrame in einen Sekundenframe eingetragen. Die Aufteilung von Meldungen in Meldungsteile erlaubt es, Standard-Meldungen zu erweitern.

Wenn z.B. die Standard-Meldung „Rotlampenfehler“ um XY erweitert werden soll, wird ein herstellereigener Meldungsteil XY definiert, der bei jeder Meldung „Rotlampenfehler“ mit übertragen wird. Damit ist es auf einfache Weise möglich, mit Filtern auf „Rotlampenfehler“ zu horchen und trotzdem herstellerspezifische Erweiterungen durchzuführen.



Schema der Handhabung von Meldungen

4.2.8 Meldungsteil

Jeder Meldungsteil wird in einem speziellen Auftragselement gespeichert. Pro Meldungsteil wird eine Auftragsnummer gespeichert, die für die Meldung selbst nicht relevant ist. Auf die Auftragsnummer folgt die Anzahl der Meldungsteile und die Member/OType-Kennung, welche die Meldungsnummer angibt. Die darauf folgende Länge des Parameterblocks ist sinnvoll, falls ein Meldungsteil aus irgendwelchen Gründen in der XML-Datei nicht bekannt ist. In einem solchen Fall könnten alle folgenden Meldungsteile und auch alle folgenden Sekundenframes/Meldungen nicht mehr ausgewertet werden. Der Mindestwert der Länge ist 4 (Länge der Vorgangskennung).

Der Aufbau der Vorgangskennung ist in 2.3 Vorgangskennung, SYSJOBID beschrieben. Es ist nicht notwendig, dass jeder Meldungsteil eine Vorgangskennung bekommt; in solchen Fällen wird eine 0 übertragen. Der Vorteil eines solchen Feldes liegt darin, dass die Zentrale in der Lage ist, bei unterschiedlichsten Meldungstypen in jedem Fall die Einträge zu finden, die zu einem bestimmten Vorgang gehören.

Die Parameter sind pro Meldung unterschiedlich und werden in der XML-Datei festgelegt, so dass sie in der Zentrale ausgewertet werden können.

Es gibt zwei verschiedene Typen von Meldungsteilen: Haupt-Meldungsteile und Zusatzmeldungsteile. Eine ‚Meldung' besteht immer aus genau einem Hauptmeldungsteil und 0..n Zusatzmeldungsteilen.

4.2.8.1 Kategorie und Schweregrad eines Meldungsteils

OCIT-Outstations definiert eine Reihe Kategorien und Schweregrade für Meldungen. Die Kategorie und der Schweregrad der Meldung sind nur am Hauptmeldungsteil festgemacht, die Zusatzmeldungsteile haben weder eine Kategorie noch einen Schweregrad.

Diese Kategorien und Schweregrade sind an zwei Punkten definiert: In der XML-Datei wird die Default-Kategorie und der Default-Schweregrad der Meldung gespeichert. Wenn projektspezifisch die Kategorie und der Schweregrad geändert werden soll, so muss das im Feldgerät zusätzlich eingetragen werden. Dies geschieht über die Methoden IncludeMT und ExcludeMT aus der Zentrale heraus.

Folgende Kategorien und Schweregrade sind definiert:

MeldungsDegree	**Beschreibung**
Information (0)	Hat keine Auswirkung auf den Verkehr.
Warnung (1)	Hat keine Auswirkungen auf den Verkehr, sollte aber bearbeitet werden.
Fehler (2)	Hat keine wesentliche Auswirkung auf den Verkehr.
Schwerer Fehler (3)	Hat wesentliche Auswirkung auf den Verkehr.

Meldungskategorie	**Beschreibung**
Sonstige (0)	Keine der folgenden Kategorien.
Geräte HW (1)	Geräte Hardware allgemein.
Sollbildfehler (2)	Signalsicherung: Steuerung versuchte fehlerhaftes Bild zu stellen.
Istbildfehler (3)	Signalsicherung: Lampenausfall in Signalgebern / Tastern, Ausbleiben oder ungewolltes Erscheinen eines Bildes.
VA (4)	Verkehrstechnische Meldungen in Anwenderprogrammen.
Übertragungssystem (5)	Kommunikation zur Zentrale (es ist zu erwarten, dass schwere Fehler dieser Kategorie nicht sofort nach oben gesendet werden).
Betriebssystem, Firmware (6)	Systemfehler und sonstiges.
Anwender SW (7)	Nicht verkehrstechnische Anwenderprogramme.
Versorgung (8)	Meldungen bei der Versorgung.
Uhr (9)	Meldungen bezogen auf Zeitfehler.

Meldungskategorie	Beschreibung
Detektoren (10)	Meldungen bezogen auf Detektoren.
Betriebszustand (11)	Betriebszustandsmeldungen.

4.2.8.2 Definition von Meldungsteilen

Meldungen werden als Liste von MELDUNGSTEILen, die erweiterte STRUCTDOMAINs sind, definiert. Der Meldungsteil ist als IdData codiert. Über die darin festgelegten MEMBER und OTYPE Werte ist ein Meldungsteil eindeutig identifiziert. Die Parameter des Meldungsteils werden als Komponenten (DECL Einträge) abgebildet.

Die einzige Einschränkung betrifft die Größe (und Anzahl) der Elemente. Es wird festgelegt, dass die Anzahl der atomaren Elemente (Zahlen oder Strings) 32 nicht überschreiten darf. Das bedeutet, dass z.B. ein Parametersatz aus 32 Simple-Domains bestehen kann oder z.B. aus einem Array mit 4 Struct-Domains mit jeweils 8 Simple-Domains usw. Referenzen werden dabei als Struct-Domains aufgefasst, bei denen die übertragenen Pfadelemente die Elemente des Structs sind.

Die Zusatzinformationen zum Meldungstyp gibt der Hersteller in den entsprechenden Feldern an:

Formatstring: Zu jedem MELDUNGSTEIL kann es einen Formatstring geben, der die Meldung kurz charakterisiert. Es ist die Verantwortung der Zentrale, diesen Formatstring auszuwerten.

MeldungsKategorie: Durch die Kategorie wird der Fehlerort oder Absender eingeschränkt.

MeldungsDegree: Grad der Einschränkung der Verkehrsregelungsfunktion des Kreuzungsgerätes

4.2.8.3 Aufbau eines Meldungsteils

Alle Meldungsteile müssen von der MSGPART MELDUNGSTEIL abgeleitet werden. Die MSGPART ist nur eine spezielle STRUCTDOMAIN, bei der drei ClassAttributes vordefiniert sind: CATEGORY, DEGREE und FORMAT. CATEGORY enthält die Meldungskategorie als Zahl, DEGREE den MeldungsDegree als Zahl und FORMAT den Formatstring.

Es stellt sich natürlich die Frage, warum ein Meldungsteil überhaupt abgeleitet wird, da ja eine Meldung ohnehin schon in Meldungsteile untergliedert ist. Die Ableitung soll auch nicht dafür verwendet werden, semantisch ähnliche Meldungsteile voneinander abzuleiten, sondern soll nur tippfaulen Leuten ersparen, die Parameter immer wieder vollständig neu hinzuschreiben, wenn semantisch unterschiedliche Meldungen deklariert werden sollen.

Wenn eine Meldung erweitert werden soll, soll nicht etwa der Meldungsteil abgeleitet werden, sondern es sollte ein neuer Meldungsteil mit dem neuen Parameter erstellt werden. Dies gilt insbesondere bei OCIT-Outstations-Meldungen. Es ist nicht zulässig, OCIT-Outstations-Meldungsteile herstellerspezifisch zu spezialisieren indem abgeleitet wird, weil dann der OCIT-Outstations-Meldungsteil in der Zentrale nicht mehr sichtbar ist!

Semantisch bedeutet die Ableitung: Die Basismeldung wird nicht zur Zentrale übertragen! Eine Ableitung schließt keine semantische Ableitung mit ein. Wenn also eine Basismeldung X existiert und eine Meldung Y von X abgeleitet wird, dann bedeutet das Übertragen von Y *nicht*, dass auch X gemeldet wurde. Stattdessen müssen neue Meldungsteile definiert werden

OCIT-Outstations definiert eine Reihe von Meldungsteilen, die weiter unten aufgeführt sind.

4.2.8.4 Formattexte für Meldungsteile

Sinn des Formattextes ist, dass eine Zentrale herstellerspezifische Meldungen eines anderen Herstellers automatisch mit Hilfe der XML-Definition des betreffenden Herstellers/Gerätes ausgeben kann. Im Formattext wird lediglich ein kurzer charakteristischer Text für die Meldung hinterlegt. Wenn die Zentrale nicht über die XML-Definitionen verfügt, gibt sie diesen Text gefolgt von den Parametern in einfacher Form aus.

Beispiel:

Formattext: Detektorstörung, Name Parameter 1 DetektorNr, Parameter 2 DetektorName.

Ausgabetext: „Detektorstörung DetektorNr=7 DetektorName=Nebenrichtung“

Es gibt Meldungen welche ein Objekt referenzieren (z.B. Signalgruppe). Falls im Meldungstext der projektspezifisch versorgte Name ausgegeben werden soll, ist dieser in der Meldung als Stringparameter zu übergeben.

4.2.9 Welche Archive existieren?

Folgende Einträge sind in OCIT-Outstations fest vorgegeben:

In jedem Feldgerät:

0 Das Betriebszustandsarchiv für die Speicherung des Betriebszustands

1 Das Standard-Meldearchiv für allgemeine Störungsmeldungen wie Fehler (Störungsarchiv)

2 Das Syslog-Archiv für projektspezifische Informationen (allgemeines Archiv)

Optional:

3 Ein Archiv für Service-Systemzugang

Zusätzlich sind zur Laufzeit von der Zentrale definierbare Messwertarchive vorgesehen. Diese haben die gleiche Struktur wie die obigen Archive und bestehen aus einer Liste von Messwertaufträgen sowie jeweils einem Ringpuffer zur Aufnahme der dynamischen Werte. Damit für Meldungen und Fehler eine definierte Speichertiefe eingehalten wird, sieht OCIT-Outstations dafür eigene Archive vor.

4.2.10 Verhalten bei Stromausfall

Für alle Listen gibt das Listenattribut CurrentPersistenz das Verhalten bei Netzausfall an. Siehe Liste:**SetSize**().

Das Betriebszustandsarchiv und das Standard-Meldearchiv bleiben erhalten (Listenauftragsstruktur und Ringpufferinhalt CurrentPersistenz=Alles=2).

4.2.11 Übertragungsformat von Archivdaten (Format der Meldung)

Jeder Meldungsteil besteht aus einer Member/OType-Kennung, die die Meldung kennzeichnet. Auf die Member/OType-Kennung folgen die Länge des Parameterblocks und die Parameter, die zu der Meldung gehören.

4.2.12 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.

Hauptmeldungsteile OCIT-Outstations (Member == 0):

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

OType	Kurzname	MeldungsDegree	Beschreibung
60000	Störung aufgehoben	I	Diese Meldung kommt, wenn eine Störung vorgelegen hat und aufgehoben wurde. Sehr häufig durch zusätzliche Meldungsteile genauer spezifiziert. Dieser Hautmeldungsteil wird nur verwendet für Meldungen für die keine spezielle Aufhebungsmeldung definiert ist.
60001	Netz aus	S	Zeigt an, zu welchem Zeitpunkt das Netz abgeschaltet wurde.
60002	Netz ein	I	Zeigt „Netzwiederkehr“ an.
60003	Systemfehler	S	Systemfehler die starke Auswirkung auf Gerätefunktion haben.
60012	Kommunikationsstörung	W	Wird eingetragen, wenn die Kommunikation mit der zentrale gestört ist.
60013	Kommunikation ok	I	Wird eingetragen, wenn die Kommunikation wieder aktiv ist.
60016	Uhr gestört	W	Wird eingetragen, wenn die Uhr gestört ist. In diesem Fall ist es eine Holschuld des Gerätes, die Uhrzeit von der Zentrale zu be-

OType	Kurzname	MeldungsDegree	Beschreibung
			schaffen, so dass die Synchronität gewahrt bleibt.
60017	Uhr ok	I	Die Uhr ist wieder in Ordnung
60018	Wartung ein	I	Die Zentrale wird informiert, dass das Gerät gewartet wird.
60019	Wartung aus	I	Die Zentrale wird informiert, dass die Wartung beendet ist.
60020	Tür auf	W	Der Türschließkontakt meldet: Die Tür des Gerätes ist offen. (Die Meldung wird nur dann verwendet, wenn projektspezifisch der Türschließkontakt vorhanden ist)
60021	Tür zu	I	Der Türschließkontakt meldet: Die Tür des Gerätes ist geschlossen. (Die Meldung wird nur dann verwendet, wenn projektspezifisch der Türschließkontakt vorhanden ist)
60026	Zeitsprung	F	Meldet das Gerät wenn es die Uhrzeit sprunghaft korrigiert. Der Zeitstempel der Meldung hat die neue Zeit. Parameter: Zeitdifferenz = Tneu – Talt als Sekunden (SLONG) Zeitquelle { Quartz, Zentrale, DCF, GPS }
60028	Suspend	I	Meldet, wenn eine Liste per Suspend angehalten wird Parameter: Listennummer.
60029	Unsuspend	I	Meldet, wenn eine Liste per Suspend angehalten wurde und per Unsuspend wieder gestartet wird Parameter: Listennummer.
60030	StartAuftrag	I	Meldet, dass ein Auftrag gestartet wurde Parameter: Listennummer, Auftragsnummer
60031	StopAuftrag	I	Meldet, dass ein Auftrag gestoppt wurde Parameter: Listennummer, Auftragsnummer
60032	ResetListe	I	Meldet, dass eine Liste RESETted wurde. Parameter: Listennummer.
60033	SyslogI	I	System Meldung Information. Parameter ist ein String.
60034	SyslogW	W	System Warnung.
60035	SyslogF	F	System Fehler

OType	Kurzname	MeldungsDegree	Beschreibung
60036	SyslogSF	S	Schwerer Systemfehler
60039	TuerAufGeraete-teil	W	Optional [2]: Geräteteil wurde geöffnet. Meldung wird als Zusatzmel-dungsteil der Meldung TürAuf protokolliert.
60040	TuerZuGeraeteteil	I	Optional: Geräteteil wurde geschlossen. Meldung wird als Zusatz-meldungsteil der Meldung TürZu protokolliert.
60041	TuerAufEVUTeil	W	Optional: EVU-Teil wurde geöffnet. Meldung wird als Zusatzmel-dungsteil der Meldung TürAuf protokolliert.
60042	TuerZuEVUTeil	I	Optional: EVU-Teil wurde geschlossen. Meldung wird als Zusatz-meldungsteil der Meldung TürZu protokolliert.
60043	TürAufBedienteil	W	Optional: Bedienteil wurde geöffnet. Meldung wird als Zusatzmel-dungsteil der Meldung TürAuf protokolliert.
60044	TürZuBedienteil	I	Optional: Bedienteil wurde geschlossen. Meldung wird als Zusatz-meldungsteil der Meldung TürZu protokolliert.
60101-60108	Sondermeldung 1 bis 8	I	Projektspezifische Sondermeldung Nummer 1 bis 8
60109-60116	Sondermeldung 9 bis 16	W	Projektspezifische Sondermeldung Nummer 9 bis 16
60117-60124	Sondermeldung 17 bis 24	F	Projektspezifische Sondermeldung Nummer 17 – 24
60125-60132	Sondermeldung 25 bis 32	S	Projektspezifische Sondermeldung Nummer 25 – 32

Bei den Meldungen Reset, Suspend und Unsuspend gibt es folgende Parameter:

Listennummer (UBYTE)	Nummer der Liste, die suspended wurde

Bei den Meldungen StartAuftrag und StopAuftrag gibt es folgende Parameter:

Listennummer (UBYTE)	Nummer der Liste, deren Auftrag bearbeitet wird.
Auftragsnummer (UBYTE)	Nummer des Auftrags, der gestartet bzw. gestoppt wurde.

[2] OType 60039 bis 60044: Die Option für die erweiterten „Tür auf-Meldungen“ kann Hardwareanpassungen der Feldgeräte nach sich ziehen.

5 Abläufe Meldung und Messwerte

5.1 Listen mit vordefinierten Aufträgen

Die folgenden Listen haben vordefinierte Aufträge, welche lediglich modifiziert werden können:

- Das Betriebszustandsarchiv für die Speicherung des Betriebszustands
- Das Standard-Meldearchiv für allgemeine Störungsmeldungen wie Fehler (Störungsarchiv)
- Das Syslog-Archiv

Bei Reset der Listen werden die vordefinierten, ursprünglichen Aufträge wiederhergestellt.

5.1.1 Ziel

Es existiert eine ‚gestartete' Liste, von der dynamische Daten abgeholt werden können.

5.1.2 Ablauf

- **Reset der Liste (400:107)**: Die Funktion Reset der Liste hat zwei Bedeutungen. Bei nicht aktivierten Listen legt sie intern die Liste an und bei bereits laufenden Listen beendet sie den aktuellen Lauf und setzt die Werte auf die Defaults zurück. Das Ergebnis ist in jedem Fall eine frisch angelegte Liste.

- **Festlegung der Listengröße (400:110)**: Jede Liste wird vom Hersteller mit einer Default-Größe ausgestattet. Diese Größe des Puffers kann mit Hilfe des Befehls geändert werden. Es ist möglich, dass für bestimmte Listen keine Größenänderung möglich ist. In diesem Fall wird dann die aktuelle Größe zurückgemeldet.

- **Festlegung der Aufträge**: Die Aufträge werden in zwei Schritten angelegt. Zunächst wird der Typ des Auftrags in der Liste angelegt. Hierzu wird der Typ übergeben und die Auftragsnummer als Funktionsergebnis zurückgegeben. Anschließend wird das so entstandene Auftrags-Objekt mit dessen eigenen Methoden parametriert. Es sei darauf hingewiesen, dass unterschiedliche Auftragselemente über ggf. unterschiedliche Parametrierungsfunktionen verfügen.

- **Anlegen eines Auftrags (400:108): Der Auftrag wird in der Liste angelegt.** Die Funktion erhält die Nummer, unter der der Auftrag angesprochen werden kann und unter der die dynamischen Daten im Ringpuffer gespeichert sind. Es ist möglich, pro Auftrag festzulegen (4xx:119), ob zusätzlich zum Eintrag des Auftragsframes in den Ringpuffer ein Event ausgelöst werden soll.

- **Konfigurieren des Auftrags**: Das Konfigurieren des Auftrags ist vom Typ des Auftrags abhängig. Messwerte bestehen in der Regel aus mehreren Auftragselementen. Diese Auftragselemente müssen ebenfalls angelegt werden.

- **Anlegen eines Auftragselements (4xx:120)**: Das Anlegen des Auftragselements geschieht für alle Auftragstypen identisch über die Funktion 120. Die Funktion gibt die Nummer des Auftragselements, das hinzugefügt wurde, zurück und erlaubt dann die Konfiguration des Auftragselements.

- **Festlegen des Event-Ziels (400:109)**: Für jede Liste, die Events auslösen soll, wird ein Event-Ziel angegeben. Dies ist im Normalfall die Zentrale, es kann aber auch ein anderes Ziel eingetragen werden. Alle Events der Liste werden dann an dieses Ziel geschickt. Wenn kein Ziel eingetragen ist, wird per Default die Zentrale verwendet. Das Festlegen löst in jedem Fall einen Event bei der Zentrale aus, so dass die Zentrale weiß, welche Listen in Gebrauch sind.

- **Start der Liste**: Die Liste wird mit dem Befehl (400:105) aktiviert.

5.2 Veränderung von Listen

Das Ziel bei einer Veränderung von Listen ist, dass die alten Daten im Ringpuffer ohne Veränderung im Ringpuffer verbleiben können und trotzdem korrekt interpretiert werden. Man muss davon ausgehen, dass ein Client nur einen Teilbereich von Daten aus der Liste ausliest. Daraus wurde folgende Designentscheidung hergeleitet: **Ein Auftrag kann nicht gelöscht werden, ohne die Liste zu stoppen und neu zu starten (wobei der Puffer komplett gelöscht wird). Es ist allerdings möglich, dass ein Auftrag deaktiviert wird, so dass sich das Datenvolumen verringert.**

Wenn eine Liste geändert werden soll, sieht das Design lediglich ein komplettes Löschen der Liste und einen neuen Aufbau der Struktur vor. Der wesentliche UseCase ist eine Redefinition der Liste. Anstelle des Auslesens der Liste und einer Umdefinition kann die Liste gleich komplett neu definiert werden, was Aufrufe sparen kann.

Die aktuelle Konfiguration der Liste lässt sich durch Get-Aufrufe der Auftragselemente feststellen. Die Get-Aufrufe liefern die Konfiguration zurück.

5.3 Wechseln des Grades (der Wichtigkeit) einzelner Meldungen

Bei Meldungsarchiven wird pro Wichtigkeit der Meldung (dem sog. MeldungsDegree) ein Auftrag verwendet. Per Default sind alle Meldungen mit einem MeldungsDegree versehen. Der MeldungsDegree einer Meldung entspricht dem MeldungsDegree des Hauptmeldungsteils. Alle anderen Meldungsteile der Meldung sind für die Bestimmung des MeldungsDegree irrelevant.

Nun kann es projektspezifisch nötig sein, dass der MeldungsDegree gewechselt werden soll. Hierfür gibt es bei den Meldungsaufträgen die Funktionen IncludeMT und ExcludeMT. Mit IncludeMT wird eine Meldung einem Auftrag zugeordnet, mit ExcludeMT wird die Meldung aus einem Auftrag entfernt. Ein Meldungsteil, der in mehr als einem Auftrag vorhanden ist,

wird zweimal in den Ringpuffer geschrieben, ein Auftrag der in keiner Liste vorhanden ist, fällt unter den Tisch.

Um einen Grad zu wechseln, sind daher zwei Befehle nötig:

- Mit IncludeMT wird dem neuen Auftrag (und damit dem neuen Grad) die Meldung zugeordnet (genauer: Es wird die Kennung des Haupt-Meldungsteils der Meldung zugeordnet).

- Mit ExcludeMT wird dem alten Auftrag (und damit dem alten Grad) die Meldung entzogen.

Es ist notwendig, erst die Meldung neu zuzuweisen und dann zu löschen, damit eine Meldung, die während des Vorgangs auftritt, nicht verloren geht.

Wenn ein Element mit ExcludeMT aus einem Auftrag entfernt wurde, kann es mit IncludeMT wieder in den Auftrag aufgenommen werden. IncludeMT führt dabei selbständig das Löschen aus der Exclude-Liste durch.

5.4 Abholen von Daten

Meldungen und Messwerte werden auf die gleiche Art abgeholt. In keinem Fall werden Daten ungefragt vom Feldgerät in die Zentrale übertragen. Die Events melden nur die Abholbedürftigkeit an. Es gibt zwei unterschiedliche Anforderungen an das Abholen von Daten:

- Kontinuierliches Abholen der Daten. Beispiel: Die Zentrale holt in jedem Fall alle Betriebsmeldungen aus dem Gerät ab. Es ist unerheblich, ob die Daten einzeln oder geblockt abgeholt werden; sie werden in jedem Fall kontinuierlich alle aus dem Gerät abgeholt (Normalfall bei der Kommunikation Zentrale – Feldgerät).

- Spontanes Abholen von Teilen des Ringpuffers tritt auf, wenn z.B. Wartungsterminals Zugriff auf die Meldungen oder die Messwerte des Ringpuffers haben möchten (Normalfall Wartungsterminal – Feldgerät).

Für das Verständnis des Ablaufes ist es wichtig zu wissen: Alle Elemente, die in dem Ringpuffer einer Liste gespeichert werden, sind durch die Kombination Positionsnummer/UTC-Zeit eindeutig bestimmt. Die Kombination tritt in der gleichen Liste höchstens einmal auf. Die Kombination aus Positionsnummer und UTC-Zeit wird im nachfolgenden als RIPID (Ringpuffer-ID) bezeichnet.

5.4.1 Kontinuierliches Abholen von Daten

Kontinuierliches Abholen von Daten wird nur von einem Gerät, im Normalfall der Zentrale, durchgeführt. Für das kontinuierliche Abholen wird die RIPID des letzten bereits abgeholten Elements benötigt. Sie wird der Funktion GetSFSince (bzw. GetSFSinceWithEvent, s.u.) als Parameter übergeben. Die Funktion gibt folgende Werte zurück:

- Eine Liste von Sekundenframes [Sekundenframe1..SekundenframeN]

- Die RIPID des ‚Sekundenframe0', der im Ringpuffer ‚vor' dem Sekundenframe1 steht.

- Die RIPID des SekundenframeN.

Wenn keine Elemente verlorengegangen sind, ist die zurückgegebene RIPID des Sekundenframe0 gleich der übergebenen RIPID. Andernfalls gingen Elemente verloren.

Es ist möglich, dass mit einem Call nicht alle neuen Elemente aus dem Ringpuffer ausgelesen werden. In diesem Fall sind noch Elemente im Puffer und es wird folgerichtig als RetCode SF_FOLLOW zurückgeliefert. Sind alle Elemente aus dem Puffer ausgelesen, wird in RetCode SF_NOFOLLOW zurückgemeldet.

Nur für das kontinuierliche Abholen ist die Funktion GetSFSinceWithEvent (400:103) gedacht. Bei dieser Funktion werden nicht nur Daten auf die gleiche Art wie bei GetSFSince abgeholt, sondern zusätzlich ein Event aktiviert, der auslöst, sobald die Liste (beginnend mit dem letzten zurückgegebenen Element) einen bestimmten Füllgrad überschreitet. Es wird also die Startposition für den EventOnFull Mechanismus markiert.

5.4.2 Spontanes Abholen von Teilen des Ringpuffers

Spontanes Abholen des Ringpuffers geschieht im Normalfall durch die Systemzugänge und Wartungsterminals. Das spontane Abholen ist dadurch gekennzeichnet, dass das Wartungsterminal nicht ‚weiß', welches Element es an welcher Position abholen soll. Das Wartungsterminal kennt hingegen den Zeitbereich, der abgeholt werden soll. Um Daten aus dem Zeitbereich abholen zu können, ist folgende Vorgehensweise sinnvoll (Z_{Start} = Startzeitpunkt des Zeitbereich; Z_{Stop} = Endzeitpunkt des Zeitbereichs):

- Das Wartungsterminal ruft die Funktion GetSFSince auf, aber mit Z_{Start}-1 und einer garantiert ungültigen Position, dem Nullvalue.

- GetSFSince gibt eine Reihe von Sekundenframes zurück. Da nur die Anzahl der Sekundenframes übergeben wird, kann entweder das letzte Element im Zeitpunkt <= Z_{Stop} sein oder > als Z_{Stop}

- Solange die Zeit des letzte Element <= Z_{Stop} ist, werden mit Hilfe der RIPID des letzten zurückgegebenen Sekundenframes weitere Sekundenframes mit GetSFSince aus dem Ringpuffer geholt.

Die Funktion GetSFSinceWithEvent ist für den Systemzugang tabu! Diese ist nur für denjenigen zulässig, auf den auch die EventDestination zeigt. Eine Filterung nach Werten ist nicht vorgesehen. Sie findet ggf. im Terminal selbst statt.

5.5 Feststellen, ob die Liste von außen (Systemzugang etc.) oder während einer Systemstörung geändert wurde

Die Zentrale (bzw. das Wartungsterminal) muss sich für jede Liste die Listenversion merken. Bei jeder Änderung der Liste, d.h. bei

- Änderung eines Auftrags,

- Änderung eines Auftragselements,

- Hinzufügen eines Auftragselements und
- Hinzufügen eines Auftrags

wird die Listenversion geändert. Die Listenversion wird stets hochgezählt, also auch nach Reset. Wenn nach einem Ausfall der Versorgungsspannung die kompletten Listeninformationen gelöscht sein sollten, wird die Version mit 0 initialisiert.

Achtung: Beim reinen Starten und Stoppen von Aufträgen bleibt die Listenversion gleich.

Die Änderung der Listenversion erfolgt erst nach dem erneuten Starten des gestoppten Auftrags.

5.6 Ändern eines Auftrags während des Laufes

Anmerkung: Die vordefinierten Aufträge können nicht geändert werden.

Um einen Auftrag während des Laufes zu ändern, sind folgende Schritte nötig:

- Der Auftrag wird von der Zentrale gestoppt (4xx:122).
- Der Auftrag wird im Gerät geändert. (Hier ändert sich die Listenversion noch nicht)
- Der Auftrag wird von der Zentrale wieder gestartet. Bei einer Änderung wird hier die Listenversion hochgezählt (4xx:121).

Sobald der Auftrag wieder gestartet ist, wird die Listenversion auf den neuen Wert gesetzt. Es ist nicht notwendig, dass ein dynamischer Wert dieses Auftrags eingetragen sein muss.

Das Feldgerät weist alle Änderungsaufträge zurück, die nicht von der EventDestination stammen, so dass die Zentrale nicht ‚befürchten' muss, dass die Listen unvermittelt von Wartungsterminals geändert werden.

5.7 Getriggertes Abholen von Daten

Es gibt drei verschiedene Typen von Triggern, die von Listen im Feldgerät zur Zentrale geschickt werden:

- OnFull bei Erreichen des Füllgrads
- OnInvalidate bei der Änderung der EventDestination
- OnInsert beim Eintragen bestimmter dynamischer Werte von ‚aktivierten' Aufträgen.

Es ist die Aufgabe der Zentrale, auf die Events angemessen zu reagieren. Die Events enthalten keine Sachdaten, sondern ‚lediglich' den Absender.

Bei OnFull und bei OnInsert müssen i.d.R. die Sekundenframes aus der entsprechenden Liste abgeholt werden.

Bei OnInvalidate hängt die Reaktion stark vom inneren Zustand der Zentrale ab. Die Änderung der EventDestination tritt sinnvollerweise bei folgenden Ereignissen auf:

- Die Zentrale ist ‚ausgefallen' und ihre Funktion wird von einer Ersatzzentrale übernommen.
- Ein anderes Gerät als die Zentrale hat sich eine eigene Liste aufgebaut und möchte diese benutzen.
- Die Zentrale delegiert die Behandlung bestimmter Listen an ein anderes Gerät.

5.8 Abholen eines Datensatzes sofort nach Auftreten

Es gibt hier zwei unterschiedliche Anforderungen, die beide abgedeckt sind: Wenn aus einer Liste nur bestimmte Aufträge sofort abgeholt werden sollen, z.B. bei Meldungen nur Fehler und schwere Fehler, werden die Aufträge aktiviert. In diesem Fall wird der Event ausgelöst, wenn der Auftrag einen Wert in den Ringpuffer schreibt.

Wenn aus einer Liste immer alle Daten abgeholt werden sollen, nachdem sie eingetragen wurden, kann auch der Füllgrad auf 0% gesetzt werden. In diesem Fall wird der Event OnFull nach jedem neu eingetragenen Datensatz ausgelöst.

5.9 Synchronisieren nach Übertragungsstörung

Wenn ein GetSFSince fehlschlägt, muss das GetSFSince einfach wiederholt werden.

Wenn ein GetSFSinceWithEvent fehlschlägt, muss entsprechend die Funktion GetSFSinceWithEvent wiederholt werden.

OCIT-O-Basis_V1.1_A02