vs | verkehrssysteme

vs | plus 10

Release notes

Cyrill Jucker

1. Oktober 2026

vs | verkehrssysteme ag | neue bahnhofstrasse 160 | ch–4132 muttenz
+41 61 501 41 41 | info@vs-plus.com | www.vs-plus.com

INHALTSVERZEICHNIS

1 VORWORT 4

2 NEUER PROJEKTAUFBAU FÜR DIE INTEGRATION DER VERSION 10 4

2.1 Struktur und Zusammenführung 4

2.2 Steuergeräteschnittstelle (Interface) 4

2.3 Funktionen und Einbindung 4

3 ERSTELLEN 6

3.1 Compiler / Sprachstandard 6

4 ERWEITERUNGEN SCHNITTSTELLE 6

4.1 OCIT-Farbcodes 6

4.2 Neu hinzugefügte Funktionen 6

4.2.1 Funktionen der Signalgruppen 6

4.2.1.1 Zuordnung eines digitalen Ausgangs zu einer Signalgruppe 6

4.2.2 Funktionen der digitalen Ausgänge 6

4.2.2.1 Ermittlung der Einschalt- und Ausschaltdauer eines digitalen Ausgangs 6

4.2.2.2 Zugriffsprüfung für digitale Ausgänge 7

4.2.2.3 Ermittlung der zugehörigen Signalgruppe eines digitalen Ausgangs 7

4.2.3 Funktionen der Bedarfssteuerung 7

4.2.3.1 Prüfung unterstützter OCIT-Farbcodes für eine Signalgruppe 7

4.2.3.2 Unterstützung der Deaktivierung von Übergängen im Bedarfsbetrieb 8

4.2.3.3 Prüfung des logischen Zustands von OCIT-Farbcodes einer Signalgruppe 8

4.2.3.4 Ansteuerung von Signalgruppen im Bedarfsbetrieb (open / closed) 9

4.2.4 Funktionen des neuen OeV-Moduls 9

4.2.4.1 Empfang von OeV-Anforderungsnachrichten (Request Messages) 9

4.2.4.2 Senden von OeV-Antwortnachrichten (Response Messages. AMLi) 10

4.3 Überarbeitete bestehende Funktionen 10

4.3.1 Geänderte Signaturen 10

4.3.1.1 Abfrage der Zwischenzeiten im Steuergerät 10

4.3.2 Umbenannte Funktionen 10

4.3.2.1 Funktionsumstellung: getSignalGroupPartialNode → getSignalGroupPartialNodeId 10

4.3.2.2 Funktionsumstellung: isSignalGroupInOffStateColour → isOnDemandColourActiveForSignalGroup 11

4.3.2.3 Funktionsumstellung: switchSignalGroupOffStateColour → switchSignalGroupOnDemandColour 11

4.3.2.4 Funktionsumstellung: getElapsedOffStateColour → getElapsedOnDemandColour 11

4.4 Entfernte Funktionen 12

4.4.1 Ungenutzte Funktionen 12

4.4.2 Relais, Blinker, digitale Ausgänge 12

4.4.3 Ersatz des alten OeV-Moduls 12

4.5 Eventuell notwendige Korrekturen auf Geräteseite 12

4.5.1 Funktionen isOpen / isClosed 12

4.5.2 Funktionen isOpenMin / isClosedMin 12

4.5.3 Funktionen getElapsedOpen / getElapsedClosed 13

4.5.4 Funktionen getElapsedOpenMin / getElapsedClosedMin 13

5 ERWEITERUNG DER SPEICHERALLOKATION 13

5.1 Erweiterung der Memory-IDs 13
5.2 Randbedingungen zur Laufzeit 13
5.3 Speicherkonzept und STL-Nutzung 14
5.4 Neue Speicher-IDs für das OeV-Modul 14
6 DAS NEUE OEV-MODUL 14
6.1 Datenstrukturen und Formate 14
6.2 OeV-Anforderungen 14
6.3 OeV-Antworten 15
6.4 AMLi-Ausgabe (Extended Serial R09) 15
7 ERWEITERUNG DER PROZESSDATEN 15
7.1 Erweiterte Prozessdatenstruktur im neuen OeV-Modul 15
7.2 Ablösung historischer Linien-Kurs-Kennungen durch Prozessdaten im OeV-Modul 15
7.3 Allgemeine Prozessdaten-Ausgabe für OeV-Anfragen 16
7.4 OeV-Prozessdaten (Extended R09 / AMLi) 17
7.5 Digitale OeV-Prozessdaten (Digital R09 / SREM/SSEM) 18
7.6 Zusätzliche OeV-Prozessdaten (CAM – Cooperative Awareness Message) 18
7.7 Zusätzliche OeV-Prozessdaten (SREM – Signal Request Extended Message) 18
7.8 Allgemeine OeV-Prozessdaten (typunabhängig) 19
8 LAUFENDE ERWEITERUNGEN UND KORREKTUREN 20
8.1 Verbesserungen aus vs | plus 9.1.0 (Patch 6) 20
8.2 Verbesserungen in Version 10.0.0 21

1 VORWORT

Version 10 wird in zwei Stufen eingeführt. Mit Version 10.0.0 steht der Funktionsumfang von 9.1.0 sowie ein vollständig erneuertes OeV-Modul inkl. SREM/CAM Unterstützung zur Verfügung. Zudem wurden Relais und Blinker als digitale Ausgänge vereinheitlicht. Die Schnittstelle zwischen vs | plus und dem Steuergerät wurde entsprechend erweitert. Als nächstes wartet die Version 10.0.1, welche v.a. die in Version 9.0 eingeführte vs | plus Bedarfssteuerung optimieren wird und Anpassungen basierend auf den Erfahrungen im verkehrstechnischen Einsatz einfliessen lässt.

Die Grundlagen für die Integration von vs | plus (9.1) entnehmen Sie dem Implementation Manual (v03-00-06). Betrachten Sie dieses Dokument als eine Weiterführung des bestehenden Manuals. Bitte achten Sie insbesondere darauf, die in Abschnitt 4.5 gestellten Anforderungen an bestehende Funktionen zu erfüllen.

2 NEUER PROJEKTAUFBAU FÜR DIE INTEGRATION DER VERSION 10

2.1 Struktur und Zusammenführung

Die gelieferte ZIP-Datei enthält auf oberster Ebene zwei Verzeichnisse (siehe dazu die nachfolgende Abbildung der Dateistruktur):

- **«Interface» (grün umrandet)**
- **«VsPlus10-<Build-Nr.>» (dunkelblau umrandet)**

Gehen Sie für die Integration wie folgt vor:

1. Erstellen Sie ein gemeinsames **Root-Verzeichnis**.
2. Kopieren Sie den gesamten Inhalt beider Ordner (inkl. aller Unterverzeichnisse) in dieses Root-Verzeichnis.
3. Behalten Sie die bestehende Verzeichnisstruktur unverändert bei.
4. Konfigurieren Sie auf Projektebene die **Include-Pfade**, sodass alle «include»-Verzeichnisse eingebunden sind.
   → Ziel: Alle Header-Dateien müssen beim Build korrekt aufgelöst werden.

Der **hellblau umrandete Bereich** enthält den vs | plus-Core.

2.2 Steuergeräteschnittstelle (Interface)

Der gesamte steuergeräterelevante Code befindet sich im **grün umrandeten «Interface»-Bereich**:

- **Orange umrandet:**
  Deklarationen der Datentypen und Strukturen für die Kommunikation zwischen Steuergerät und vs | plus
- **Rot umrandet:**
  - Konfigurationsdaten (steuergerätespezifisch)
  - Deklarationen der funktionalen Schnittstelle

Wichtig: Die Konfigurationsdaten müssen entsprechend dem Ziel-Steuergerät angepasst und ergänzt werden.

2.3 Funktionen und Einbindung

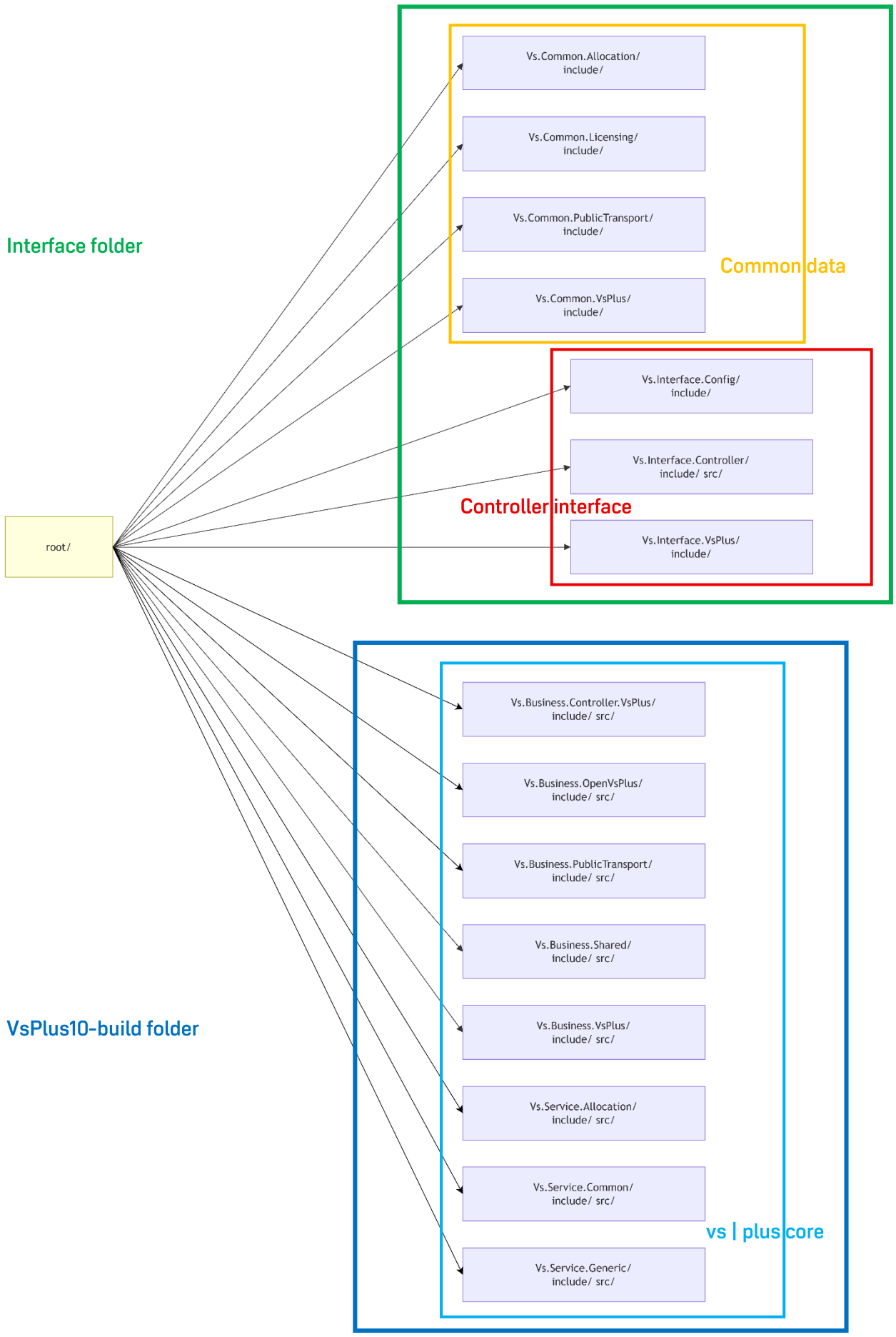
Die Schnittstellenfunktionen wurden in Version 10 erweitert:

- Neue Funktionen müssen gemäss Anleitung und Inline-Kommentaren implementiert werden
- Die in 4.5 gelisteten Funktionen sind zu prüfen und ggf. anzupassen

Relevante Dateien:

- **Einbindung Steuergerät → vs | plus:** VsplusControllerFunctions.c
- **Von vs | plus bereitgestellte Funktionen:** VsplusFunctions.h

Ihre Anpassungen finden im **Interface-Bereich** statt, während der vs | plus Core unverändert bleibt.


Interface folder
Vs.Common.Allocation/
include/
Vs.Common.Licensing/
include/
Vs.Common.PublicTransport/
include/
Vs.Common.VsPlus/
include/
Common data
Vs.Interface.Config/
include/
Vs.Interface.Controller/
include/ src/
Controller interface
Vs.Interface.VsPlus/
include/
root/
VsPlus10-build folder
Vs.Business.Controller.VsPlus/
include/ src/
Vs.Business.OpenVsPlus/
include/ src/
Vs.Business.PublicTransport/
include/ src/
Vs.Business.Shared/
include/ src/
Vs.Business.VsPlus/
include/ src/
Vs.Service.Allocation/
include/ src/
Vs.Service.Common/
include/ src/
vs | plus core
Vs.Service.Generic/
include/ src/

3 ERSTELLEN

3.1 Compiler / Sprachstandard

Die Erstellung von Version 10 erfolgt in C/C++ und setzt mindestens den Sprachstandard **C++11** voraus. Das neue **OeV-Modul** ist überwiegend in C++ implementiert und basiert ebenfalls auf diesem Mindeststandard.

Alle übrigen bestehenden Codebereiche sind weiterhin in **C** umgesetzt.

4 ERWEITERUNGEN SCHNITTSTELLE

4.1 OCIT-Farbcodes

Die aktuell von der vs | plus Bedarfssteuerung unterstützten OCIT Farbcodierungen (Hex-Werte) entnehmen Sie bitte der Enumeration in OcitColorCodes.h (Vs.Common.VsPlus), gemäss OCIT-Standard.

4.2 Neu hinzugefügte Funktionen

4.2.1 Funktionen der Signalgruppen

4.2.1.1 Zuordnung eines digitalen Ausgangs zu einer Signalgruppe

Element	Beschreibung
Funktionsname	getSignalGroupChannelRelatedDigitalOutput
Signatur	signed short getSignalGroupChannelRelatedDigitalOutput(signed short sg)
Kurzbeschreibung	Ermittelt den Kanal des digitalen Ausgangs, der der angegebenen Signalgruppe zugeordnet ist (gemäss OCIT-Zuordnung).
Parameter	sg – Kanalnummer der Signalgruppe
Rückgabewert	Kanalnummer des zugeordneten digitalen Ausgangs
Fehlerfall	Gibt 0 zurück, wenn die angegebene Signalgruppe ungültig ist oder kein zugehöriger digitaler Ausgang existiert

4.2.2 Funktionen der digitalen Ausgänge

4.2.2.1 Ermittlung der Einschalt- und Ausschaltdauer eines digitalen Ausgangs

Element	Beschreibung
Funktionsnamen	getElapsedDigitalOn / getElapsedDigitalOff
Signaturen	unsigned short getElapsedDigitalOn(signed short port) unsigned short getElapsedDigitalOff(signed short port)
Kurzbeschreibung	Ermitteln die verstrichene Zeit seit dem letzten Ein- bzw. Ausschalten eines digitalen Ausgangs.
Parameter	port – Kanalnummer des digitalen Ausgangs
Rückgabewert	Verstrichene Zeit seit Zustandswechsel in **Dezisekunden**
Verhalten (On)	Gibt die Zeit seit dem Einschalten zurück; Rückgabewert 0, wenn der Kanal ungültig ist oder der Ausgang aktuell ausgeschaltet ist
Verhalten (Off)	Gibt die Zeit seit dem Ausschalten zurück; Rückgabewert 0, wenn der Kanal ungültig ist oder der Ausgang aktuell eingeschaltet ist

4.2.2.2 Zugriffsprüfung für digitale Ausgänge

Element	Beschreibung
Funktionsname	isDigitalOutputEnabledForVsplus
Signatur	unsigned short isDigitalOutputEnabledForVsplus(signed short port)
Kurzbeschreibung	Prüft, ob ein digitaler Ausgang für den Zugriff durch vs | plus freigegeben ist. Analog zur Zugriffskontrolle bei Signalgruppen.
Parameter	port – Kanalnummer des digitalen Ausgangs
Rückgabewert	Wert > 0: Zugriff erlaubt Wert = 0: Kanal ungültig oder Zugriff nicht erlaubt

4.2.2.3 Ermittlung der zugehörigen Signalgruppe eines digitalen Ausgangs

Element	Beschreibung
Funktionsname	getDigitalOutputRelatedSignalGroupChannel
Signatur	signed short getDigitalOutputRelatedSignalGroupChannel(signed short port)
Kurzbeschreibung	Ermittelt die zur angegebenen digitalen Ausgangsleitung gehörende Signalgruppe gemäss OCIT-Zuordnung.
Parameter	port – Kanalnummer des digitalen Ausgangs
Rückgabewert	Kanalnummer der zugehörigen Signalgruppe
Fehlerfall	Gibt 0 zurück, wenn der Kanal ungültig ist oder keine zugehörige Signalgruppe existiert

4.2.3 Funktionen der Bedarfssteuerung

4.2.3.1 Prüfung unterstützter OCIT-Farbcodes für eine Signalgruppe

Element	Beschreibung
Funktionsname	areOcitColourCodesSupported
Signatur	unsigned short areOcitColourCodesSupported(signed short signalGroupChannel, OcitColourCodes* codes, int count)
Kurzbeschreibung	Prüft, ob die übergebenen OCIT-Farbcodes für die angegebene Signalgruppe konfiguriert und verfügbar sind.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe codes – Array von OCIT-Farbcodes count – Anzahl der Elemente im Array
Rückgabewert	Wert > 0: Alle Farbcodes sind verfügbar
Fehlerfall	Gibt 0 zurück, wenn die Signalgruppe ungültig ist oder mindestens ein Farbcode nicht unterstützt wird

4.2.3.2 Unterstützung der Deaktivierung von Übergängen im Bedarfsbetrieb

Element	Beschreibung
Funktionsname	canDisableTransitionsWithinOnDemandControl
Signatur	unsigned short canDisableTransitionsWithinOnDemandControl(void)
Kurzbeschreibung	Prüft, ob der Controller im Bedarfsbetrieb (on-demand control) das Deaktivieren von Zustandsübergängen unterstützt.
Parameter	—
Rückgabewert	Wert > 0: Deaktivierung von Übergängen ist möglich. Das Steuergerät muss in diesem Fall für alle Signalgruppen Übergänge wahlweise schalten oder unterdrücken können.
Fehlerfall	Gibt 0 zurück, wenn das Deaktivieren von Übergängen nicht oder nur teilweise unterstützt wird.

4.2.3.3 Prüfung des logischen Zustands von OCIT-Farbcodes einer Signalgruppe

Element	Beschreibung
Funktionsnamen	isOcitColorCodeSignalGroupLogicalStateOpen / isOcitColorCodeSignalGroupLogicalStateClosed
Signaturen	signed short isOcitColorCodeSignalGroupLogicalStateOpen(signed short signalGroupChannel, OcitColourCodes ocitColourCode) signed short isOcitColorCodeSignalGroupLogicalStateClosed(signed short signalGroupChannel, OcitColourCodes ocitColourCode)
Kurzbeschreibung	Prüfen, ob der logische Zustand eines bestimmten OCIT-Farbcodes einer Signalgruppe dem Zustand «open» bzw. «closed» entspricht.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe ocitColourCode – OCIT-Farbcode (Enum-Wert)
Rückgabewert (Open)	Wert > 0: logischer Zustand ist «open» Wert = 0: Signalgruppe oder Farbcode ungültig oder Zustand ist «closed»
Rückgabewert (Closed)	Wert > 0: logischer Zustand ist «closed» Wert = 0: Signalgruppe oder Farbcode ungültig oder Zustand ist «open»
Einschränkung	Der Zustand «undefined» wird derzeit nicht unterstützt. Rückgabe des Werts 0.

4.2.3.4 Ansteuerung von Signalgruppen im Bedarfsbetrieb (open / closed)

Element	Beschreibung
Funktionsnamen	switchSignalGroupOpenByOnDemandControl / switchSignalGroupClosedByOnDemandControl
Signaturen	void switchSignalGroupOpenByOnDemandControl(signed short signalGroupChannel, int doIncludeTransition) void switchSignalGroupClosedByOnDemandControl(signed short signalGroupChannel, int doIncludeTransition)
Kurzbeschreibung	Fordert das Umschalten einer Signalgruppe in den Zustand «open» bzw. «closed» während eines aktiven Bedarfsbetriebs (on-demand control) an.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe doIncludeTransition – Gibt an, ob ein Übergang der Grundversorgung («open->closed», bzw. «closed->open») geschaltet werden soll oder nicht
Verhalten (Open)	Schaltet die Signalgruppe in den Zustand «open». Bei gesetztem Parameter wird ein Übergang von «closed» nach «open» ausgeführt.
Verhalten (Closed)	Schaltet die Signalgruppe in den Zustand «closed». Bei gesetztem Parameter wird ein Übergang von «open» nach «closed» ausgeführt.
Einschränkung	Die Funktionen werden ausschließlich während eines aktiven Bedarfsbetriebs aufgerufen

4.2.4 Funktionen des neuen OeV-Moduls

4.2.4.1 Empfang von OeV-Anforderungsnachrichten (Request Messages)

Element	Beschreibung
Funktionsname	readPublicTransportRequestMessagePackage
Signatur	signed short readPublicTransportRequestMessagePackage(PublicTransportMessagePackage* const message)
Kurzbeschreibung	Liest ein OeV-Anforderungsnachrichtenpaket aus der Nachrichtenwarteschlange des Controllers.
Parameter	message – Ausgabestruktur, wird mit Zeiger auf den Nachrichteninhalt sowie dem Nachrichtentyp (z. B. CAM, SREM, Serial R09) befüllt
Rückgabewert	Wert > 0: Neues Nachrichtenpaket erfolgreich gelesen
Fehlerfall	Gibt 0 zurück, wenn keine neue Nachricht verfügbar ist oder das Paket nicht gesetzt werden konnte

4.2.4.2 Senden von OeV-Antwortnachrichten (Response Messages. AMLi)

Element	Beschreibung
Funktionsname	sendPublicTransportMessageResponsePackages
Signatur	void sendPublicTransportMessageResponsePackages(const PublicTransportMessageResponsePackage* const packages, int count)
Kurzbeschreibung	Überträgt ein Array von OeV-Antwortnachrichten an den Controller. Die Antworten beziehen sich auf zuvor gelesene Anfragen.
Parameter	packages – Array von Antwortpaketen (z. B. SerialR09 (AMLi)-, CAM- oder SREM-Antworten inkl. Typkennzeichnung) count – Anzahl der übergebenen Pakete
Rückgabewert	—

4.3 Überarbeitete bestehende Funktionen

4.3.1 Geänderte Signaturen

4.3.1.1 Abfrage der Zwischenzeiten im Steuergerät

Element	Beschreibung
Funktionsname	getControllerIntergreen
Signatur	signed short getControllerIntergreen(signed short sgclear, signed short sgenter)
Kurzbeschreibung	Liefert die Zwischenzeit (Intergreen) zwischen zwei Signalgruppen gemäss aktuellem Signalprogramm in der Grundversorgung des Steuergeräts.
Parameter	sgclear – Kanalnummer der räumenden Signalgruppe
	sgenter – Kanalnummer der einfahrenden Signalgruppe
Rückgabewert	Zwischenzeit in Dezisekunden
Fehlerfall	Ungültige Signalgruppe(n) – maximaler signed short-Wert
	Keine Zwischenzeit definiert – maximaler signed short-Wert

4.3.2 Umbenannte Funktionen

4.3.2.1 Funktionsumstellung: getSignalGroupPartialNode → getSignalGroupPartialNodeId

Element	Beschreibung
Funktionsname	getSignalGroupPartialNodeId
Signatur	unsigned short getSignalGroupPartialNodeId(signed short signalGroupChannel)
Kurzbeschreibung	Liefert die ID des verkehrstechnischen Teilknotens, dem die Signalgruppe zugeordnet ist.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe
Rückgabewert	Gültige Teilknoten-ID [1..max]; bei ungültiger oder nicht zugeordneter Signalgruppe: 0

4.3.2.2 Funktionsumstellung: isSignalGroupInOffStateColour → isOnDemandColourActiveForSignalGroup

Element	Beschreibung
Funktionsname	isOnDemandColourActiveForSignalGroup
Signatur	unsigned short isOnDemandColourActiveForSignalGroup(signed short signalGroupChannel, OcitColourCodes ocitColourCode)
Kurzbeschreibung	Prüft, ob eine Signalgruppe aktuell die angegebene Farbe aktiv hat.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe
	ocitColourCode – OCIT-Farbcode
Rückgabewert	> 0 wenn aktiv; 0 bei ungültigen Parametern oder anderer aktueller Farbe

4.3.2.3 Funktionsumstellung: switchSignalGroupOffStateColour → switchSignalGroupOnDemandColour

Element	Beschreibung
Funktionsname	switchSignalGroupOnDemandColour
Signatur	void switchSignalGroupOnDemandColour(signed short signalGroupChannel, OcitColourCodes ocitColourCode, int doIncludeTransition)
Kurzbeschreibung	Fordert das Steuergerät auf, eine Signalgruppe auf eine bestimmte Farbe zu schalten.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe
	ocitColourCode – OCIT-Farbcode
	doIncludeTransition – Übergangssequenz berücksichtigen
Rückgabewert	—

4.3.2.4 Funktionsumstellung: getElapsedOffStateColour → getElapsedOnDemandColour

Element	Beschreibung
Funktionsname	getElapsedOnDemandColour
Signatur	unsigned short getElapsedOnDemandColour(signed short signalGroupChannel, OcitColourCodes ocitColourCode)
Kurzbeschreibung	Liefert die vergangene Zeit seit dem Umschalten der Signalgruppe auf die angegebene Farbe.
Parameter	signalGroupChannel – Kanalnummer der Signalgruppe
	ocitColourCode – OCIT-Farbcode
Rückgabewert	Zeit in Dezisekunden
Fehlerfall	Ungültige Parameter oder andere Farbe aktiv – Rückgabewert 0

4.4 Entfernte Funktionen

4.4.1 Ungenutzte Funktionen

Folgende Funktionen wurden seit längerem nicht mehr von der Steuergerätschnittstelle unterstützt, bzw. von vs | plus verwendet und sind deshalb entfernt worden:

- signed short getDetSSCounter(const signed short det)
- void clearDetSSCounter(const signed short det)

4.4.2 Relais, Blinker, digitale Ausgänge

Durch die Zusammenführung der Ports «Relais» und «Blinker» zu digitalen Ausgängen entfallen die bisherigen Blinker-Funktionen und wurden aus der Schnittstelle entfernt:

- void switchOnBlinker(const signed short channel)
- void switchOffBlinker(const signed short channel)
- signed short isOnBlinker(const signed short channel)
- signed short isOffBlinker(const signed short channel)

4.4.3 Ersatz des alten OeV-Moduls

Das neue OeV-Modul ermöglicht den Empfang verschiedener OeV-Nachrichten, bzw. Telegramme. Die Funktion, welche die reine serielle R09 Übertragung angeboten hatte, wurde deshalb entfernt:

- signed short readPTMessage(void *const msg)

4.5 Eventuell notwendige Korrekturen auf Geräteseite

Die nachfolgenden Implementierungen der Funktionen zur Bereitstellung der Signalgruppendaten müssen überprüft und bei Bedarf korrigiert werden, sofern sie nicht den nachfolgend definierten Anforderungen entsprechen.

Standardzustand entspricht dem in der Grundversorgung definierten Zustand **«Open»** , bzw. **«Closed»** («Frei», bzw. «Sperren») einer Signalgruppe.

4.5.1 Funktionen isOpen / isClosed

Die Zustände **«Open»** und **«Closed»** («Frei» und «Sperren») beziehen sich ausschließlich auf die jeweiligen **Standardzustände und -farben** der Signalgruppen.

Abweichende Betriebszustände, insbesondere solche aus der Bedarfssteuerung (z. B. «gelb blinkend»), sind hierbei entsprechend zu berücksichtigen:
Eine Farbe, die funktional den Zustand «Open» signalisiert, aber **nicht der definierten Standardfarbe für «Open» entspricht**, darf von der Funktion **«isOpen» nicht als «wahr»** interpretiert werden.

Die Funktion **«isOpen»** darf nur dann den Wert «wahr» liefern, wenn die aktive Farbe tatsächlich dem spezifizierten **Standardzustand «Open»** entspricht.

4.5.2 Funktionen isOpenMin / isClosedMin

Die Funktionen **«isOpenMin»** und **«isClosedMin»** müssen anzeigen, ob sich die Signalgruppe für mindestens die definierte **Mindestdauer** im jeweiligen **Standardzustand** befindet.

Analog zu «isOpen» und «isClosed» beziehen sich diese Funktionen ausschließlich auf die festgelegten **Standardzustände und -farben**.

Ist eine Farbe (z. B. «gelb blinkend») als **Standardzustand** definiert, so ist der Zeitraum bei der Prüfung der Mindestdauer vollständig zu berücksichtigen.

Befindet sich die Signalgruppe in einem **abweichenden Zustand oder einer Farbe**, der nicht dem definierten Standard entspricht, darf dieser Zeitraum **nicht** in die Mindestdauer einfließen; die Zeitmessung ist in diesem Fall **zurückzusetzen**.

Die Funktionen dürfen nur dann den Wert «wahr» liefern, wenn die Signalgruppe die Farbe des Standard-Frei- bzw. des Standard-Sperren-Zustandes **ohne Unterbruch** innerhalb der definierten Mindestzeit erfüllt.

4.5.3 Funktionen getElapsedOpen / getElapsedClosed

Die Funktionen **«getElapsedOpen»** und **«getElapsedClosed»** müssen die Dauer zurückgeben, während der sich die Signalgruppe im jeweiligen **Standardzustand** befindet.

Analog zu den Zuständen «isOpen» und «isClosed» beziehen sich diese Funktionen ausschließlich auf die definierten **Standardzustände und -farben**.

Ist ein Zustand wie z. B. «gelb blinkend» als **Standardzustand** definiert, so ist die Zeitdauer entsprechend fortzuschreiben und zurückzugeben.

Befindet sich die Signalgruppe hingegen in einem **abweichenden Zustand oder einer Farbe**, der nicht dem definierten Standard entspricht, darf die Zeit **nicht weiter fortgeschrieben** werden; in diesem Fall ist die Zeitmessung **zurückzusetzen**.

Die zurückgegebene Dauer ist daher ausschließlich für Zeiträume gültig, in denen die aktive Signalphase tatsächlich dem spezifizierten **Standardzustand** entspricht.

4.5.4 Funktionen getElapsedOpenMin / getElapsedClosedMin

Die Funktionen **«getElapsedOpenMin»** und **«getElapsedClosedMin»** müssen die aktuell akkumulierte Zeitdauer zurückgeben, während der sich die Signalgruppe im jeweiligen **Standardzustand im Hinblick auf die Mindestzeit** befindet.

Die Zeitmessung bezieht sich ausschließlich auf **zusammenhängende Zeitintervalle** im definierten Standardzustand.

Ist ein Zustand (z. B. «gelb blinkend») als **Standardzustand** definiert, so wird die Zeit für diesen Zustand entsprechend fortgeschrieben.

Befindet sich die Signalgruppe hingegen in einem **abweichenden Zustand oder einer Farbe**, der nicht dem definierten Standard entspricht, darf die Zeit **nicht weiter fortgeschrieben** werden; in diesem Fall ist die Zeitmessung **zurückzusetzen**.

Die zurückgegebene Dauer dient als Grundlage zur Bewertung, wie weit die jeweilige **Mindestzeit fortgeschritten ist**.

5 ERWEITERUNG DER SPEICHERALLOKATION

5.1 Erweiterung der Memory-IDs

In der Datei **MemoryIdentifiers.h** (Vs.Common.Allocation) wurden neue IDs für Speicherbereiche hinzugefügt. Seit Version 9 verwendet vs | plus einen internen Memory Manager, der bis zu **500 Speicherbereiche** beim Steuergerät verwalten kann. Die möglichen IDs reichen dabei über den gesamten Integer-Wertebereich (min bis max int).

5.2 Randbedingungen zur Laufzeit

Das neue OeV-Modul ist in C++ entwickelt und nutzt die **STL (Standard Template Library)**. Dabei gilt eine zentrale Einschränkung: Nach der Initialisierung von vs | plus darf **kein Speicher mehr dynamisch angefordert oder freigegeben werden.**

5.3 Speicherkonzept und STL-Nutzung

Um den Einsatz von STL-Containern dennoch zu ermöglichen, wird bereits beim Start **ein größerer Speicherblock** reserviert.

- Zugriff erfolgt über spezielle **Allokatoren**
- Nutzung des vorab reservierten Speichers mittels **placement new/delete**
- Keine dynamischen Speicheroperationen zur Laufzeit notwendig

Dieses Konzept ermöglicht eine effiziente Nutzung der STL unter den gegebenen Systemrestriktionen. Das OeV-Modul ist das erste Modul in vs | plus, das dieses Prinzip verwendet. Aktuell sind dafür **1 MB (1.048.576 Bytes)** reserviert.

5.4 Neue Speicher-IDs für das OeV-Modul

Für das OeV-Modul wurden folgende IDs definiert:

- MEMORY_POOL = 0x7FFFFFFF
- MEMORY_POOL_FREE_LISTS = 0x7FFFFFFE
- MEMORY_POOL_INTERNAL = 0x7FFFFFFD
- PUBLIC_TRANSPORT_DATA_PROVIDER = 0x7FFFFFFC
- PUBLIC_TRANSPORT_PROCESSOR = 0x7FFFFFFB
- PUBLIC_TRANSPORT_RUNNER = 0x7FFFFFFA

6 DAS NEUE OEV-MODUL

Mit Version 10 wird das bisherige OeV-Modul vollständig durch eine neue Implementierung ersetzt. Das alte Modul basierte auf einer Third-Party-Lösung, die über einen längeren Zeitraum zuverlässig eingesetzt wurde, jedoch nicht mehr den aktuellen technologischen und konzeptionellen Anforderungen entspricht.

Die neue Lösung wurde komplett in C++ (mindestens C++11) entwickelt und nutzt dessen moderne Sprachmerkmale zur Verbesserung von Wartbarkeit, Erweiterbarkeit und Codequalität.

Die bisherige Funktionalität wurde übernommen, konzeptionell überarbeitet und gezielt modernisiert. Das alte Modul wird nicht mehr unterstützt.

6.1 Datenstrukturen und Formate

Alle notwendigen Datenstrukturen und Definitionen u.a. auch für SREM- und CAM-Nachrichten sind im Unterprojekt **Vs.Common.PublicTransport** enthalten. Sie entsprechen der jeweils gültigen OCIT-Spezifikation.

6.2 OeV-Anforderungen

Während eines Verarbeitungszyklus ruft vs | plus mehrfach (aktuell bis zu fünfmal) die Funktion readPublicTransportRequestMessagePackage auf.

Liegen OeV-Anforderungen vor (serielles R09, CAM oder SREM), werden diese vorab in strukturierter Form als **Pakete** bereitgestellt.

Ein Paket enthält:

- genau eine gültige Anforderung
- den zugehörigen Typ (R09, CAM oder SREM) der Anforderung

6.3 OeV-Antworten

Antworten sendet vs | plus im Push-Verfahren mittels Funktion sendPublicTransportMessageResponsePackages.

Das Steuergerät entscheidet dabei selbst, ob es vorhandene Antworten ausliest. Zur Effizienzsteigerung werden ausschließlich strukturierte Werte (keine Strings mehr) übertragen.

Auch hier gilt:

- Ein Paket enthält genau eine Antwort
- inklusive zugehörigem Typ (R09 (AMLi), CAM oder SREM)

6.4 AMLi-Ausgabe (Extended Serial R09)

Das frühere Konzept eines OeV-internen Speichers (bis zu 999 AMLi-Datensätze) wurde verworfen.

Stattdessen wird:

- pro OeV-Anforderung maximal ein AMLi-Datensatz erzeugt
- dieser direkt über die Geräteschnittstelle ausgegeben (siehe. OeV-Antworten)

Eine Zwischenspeicherung und ein späteres Auslesen über direkten Zugriff entfallen damit vollständig.

7 ERWEITERUNG DER PROZESSDATEN

7.1 Erweiterte Prozessdatenstruktur im neuen OeV-Modul

Mit der Einführung des neuen OeV-Moduls wurden die verfügbaren Prozessdaten deutlich erweitert.

Eine vollständige C-style Liste aller Prozessdaten-IDs sowie der maximal zugeordneten Topologie Objekte ist in **ProcessDataList.h** (Vs.Common.VsPlus) definiert.

Die detaillierten Beschreibungen der einzelnen Prozessdaten befinden sich im selben Unterprojekt in der Datei **ProcessDataDefinitions.h**.

Der Abruf der neu verfügbaren Prozessdaten erfolgt gleichfalls über die schon bestehende und bekannte Schnittstellenfunktion. Als Instanz-ID wird jeweils die logische Detektor- / Meldepunkt-, bzw. Verkehrsstrom-ID erwartet.

7.2 Ablösung historischer Linien-Kurs-Kennungen durch Prozessdaten im OeV-Modul

Name	Beschreibung
OITD_OEV_MODIFIKATION_VS	Ersetzt die frühere Linien-Kurs-Kennung 88888 für eine OeV-Modifikation
OITD_PTMODULE_EMERGENCY_CHECKIN_ACTIVE	Ersetzt die frühere Linien-Kurs-Kennung 77777 für Notanforderungen
OITD_OEV_NOTAN_WARTEZEIT_VS	Ergänzt die Ablösung der Kennung 77777 (Notanforderung), z. B. zur Abbildung der Wartezeit
OITD_OEV_ZWANGSABM_VS	Ersetzt die frühere Linien-Kurs-Kennung 99999 für Zwangsabmeldung bzw. Zwangslöschung

7.3 Allgemeine Prozessdaten-Ausgabe für OeV-Anfragen

Name	Beschreibung
OITD_PUBLIC_TRANSPORT_REQUEST_CHANNEL	OCIT-Outstation (STG) Kanal des Meldepunkts
OITD_PUBLIC_TRANSPORT_REQUEST_TYPE	Typ der übermittelten OeV-Anfrage (PublicTransportRequestTypes)
OITD_PUBLIC_TRANSPORT_NOF_REQUESTS	Anzahl der übermittelten OeV-Anfragen im aktuellen vs | plus Zyklus
OITD_PUBLIC_TRANSPORT_REQUEST_YEAR	Jahr des OeV-Anfrage Zeitstempels [0..99]
OITD_PUBLIC_TRANSPORT_REQUEST_MONTH	Monat des OeV-Anfrage Zeitstempels [1..12]
OITD_PUBLIC_TRANSPORT_REQUEST_DAY	Tag des OeV-Anfrage Zeitstempels [1..31]
OITD_PUBLIC_TRANSPORT_REQUEST_HOUR	Stunde des OeV-Anfrage Zeitstempels [0..23]
OITD_PUBLIC_TRANSPORT_REQUEST_MINUTE	Minute des OeV-Anfrage Zeitstempels [0..59]
OITD_PUBLIC_TRANSPORT_REQUEST_SECOND	Sekunde des OeV-Anfrage Zeitstempels [0..59]

7.4 OeV-Prozessdaten (Extended R09 / AMLi)

Name	Beschreibung
OITD_R09EXT_TIMESTAMP_YEAR	Jahr des R09 Zeitstempels [0..99]
OITD_R09EXT_TIMESTAMP_MONTH	Monat des R09 Zeitstempels [1..12]
OITD_R09EXT_TIMESTAMP_DAY	Tag des R09 Zeitstempels [1..31]
OITD_R09EXT_TIMESTAMP_HOUR	Stunde des R09 Zeitstempels [0..23]
OITD_R09EXT_TIMESTAMP_MINUTE	Minute des R09 Zeitstempels [0..59]
OITD_R09EXT_TIMESTAMP_SECOND	Sekunde des R09 Zeitstempels [0..59]
OITD_R09EXT_CALLPOINT_NR	Meldepunktnummer des R09 Eingangs [1..16777216]
OITD_R09EXT_TIMETABLE_DEVIATION	Fahrplanabweichung in Sekunden [-3599..3599]
OITD_R09EXT_LINE	Liniennummer des Fahrzeugs [0..999]
OITD_R09EXT_ROUTE	Routennummer des Fahrzeugs [0..999]
OITD_R09EXT_TOUR	Kursnummer des Fahrzeugs [0..99]
OITD_R09EXT_PRIORITY	Priorität des Fahrzeugs [0..7]
OITD_R09EXT_TRAIN_LENGTH	Zuglänge des Fahrzeugs [0..7]
OITD_R09EXT_MANUAL_DIRECTION	Manuelle Richtungsangabe des Fahrzeugs [0..3]
OITD_R09EXT_RELATIVE_INTERSECTION	Relative Knotenkennung [0..255]
OITD_R09EXT_IS_MODIFICATION_BY_CENTER	Gibt an, ob eine Modifikation durch die Zentrale erfolgt [0 (nein), 1 (ja), 255 (undefiniert)]
OITD_R09EXT_RECEPTION_CYCLE_SECOND	Umlaufsekunde des R09 Empfangs [1..255]
OITD_R09EXT_SIGNAL_PLAN	Aktuell laufender Signalplan / Programm [0..32]
OITD_R09EXT_CURRENT_PHASE	Aktuelle Phase der zugeordneten Signalgruppe [0 (keine) .. 255]
OITD_R09EXT_DESIRED_PHASE	Gewünschte Phase der zugeordneten Signalgruppe [0 (keine Übergang aktiv / keine Phasen) .. 255]
OITD_R09EXT_TRAVEL_TIME	Reisezeit des Fahrzeugs [0 (nicht verfügbar / Notanforderung) .. 255]
OITD_R09EXT_SIGNAL_GROUP_OPEN_STARTED	Rotende der zugeordneten Signalgruppe [0..255]
OITD_R09EXT_SIGNAL_GROUP_OPEN_STOPPED	Grünende der zugeordneten Signalgruppe [0..255]

7.5 Digitale OeV-Prozessdaten (Digital R09 / SREM/SSEM)

Name	Beschreibung
OITD_R09DI_REPORTING_POINT	Meldepunktnummer des digitalen R09 Empfangs [0..65535]
OITD_R09DI_PRIORITY_LEVEL	Priorität des Fahrzeugs [0..255]
OITD_R09DI_LENGTH	Zuglänge des Fahrzeugs [0..7]
OITD_R09DI_ROUTE	Routennummer des Fahrzeugs [0..65535]
OITD_R09DI_LINE	Liniennummer des Fahrzeugs [0..65535]
OITD_R09DI_DIRECTION	Manuelle Richtungsangabe des Fahrzeugs [0..255]
OITD_R09DI_TOUR	Kursnummer des Fahrzeugs [0..4294967295]
OITD_R09DI_VERSION	Basisversion der OeV-Versorgung [0..4294967295]

7.6 Zusätzliche OeV-Prozessdaten (CAM – Cooperative Awareness Message)

Name	Beschreibung
OITD_CAM_STATION_ID	ID des Anforderers / Fahrzeugs
OITD_CAM_STATION_TYPE	Typ des Anforderers / Fahrzeugs
OITD_CAM_POSITION_LATITUDE	Position (Breitengrad) des Anforderers / Fahrzeugs
OITD_CAM_POSITION_LONGITUDE	Position (Längengrad) des Anforderers / Fahrzeugs
OITD_CAM_EMBARCATION_STATUS	Anforderer / Fahrzeug hat Türen geöffnet (Ja / Nein)

7.7 Zusätzliche OeV-Prozessdaten (SREM – Signal Request Extended Message)

Name	Beschreibung
OITD_SREM_STATION_ID	ID des Anforderers / Fahrzeugs
OITD_SREM_BASIC_VEHICLE_ROLE	Typ des Anforderers / Fahrzeugs
OITD_SREM_TRANSIT_SCHEDULE	Fahrplanabweichung in Sekunden (*10) [-122..121] (-122 = nicht verfügbar)
OITD_SREM_TIMESTAMP	Minute des aktuellen Jahres (UTC) [0..527040] (527040 = ungültig)
OITD_SREM_TIMESTAMP_SECOND	Millisekunde innerhalb der aktuellen UTC-Minute [0..65535]
OITD_SREM_HEADING	Richtung des Anforderers / Fahrzeugs
OITD_SREM_POSITION_LATITUDE	Position (Breitengrad) des Anforderers / Fahrzeugs
OITD_SREM_POSITION_LONGITUDE	Position (Längengrad) des Anforderers / Fahrzeugs
OITD_SREM_SPEED	Geschwindigkeit des Anforderers / Fahrzeugs

7.8 Allgemeine OeV-Prozessdaten (typunabhängig)

Name	Beschreibung
OITD_PTMODULE_CALLPOINT	Meldepunktnummer unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_VEHICLE_ID	Fahrzeug-ID (Anforderer) unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_VEHICLE_TYPE	Fahrzeugtyp (Anforderer) unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_VEHICLE_LENGTH	Zuglänge unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_PRIORITY	Priorität unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_TIMETABLE_STATUS	Fahrplanlage unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_LINE	Linie unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_TOUR	Kurs unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_ROUTE	Route unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_DIRECTION	Manuelle Richtung unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_EMERGENCY_CHECKIN_ACTIVE	Quittierung der Notanforderung (für die Synoptik)
OITD_PTMODULE_POSITION_LATITUDE	Position (Breitengrad) unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_POSITION_LONGITUDE	Position (Längengrad) unabhängig vom Typ der OeV-Anfrage
OITD_PTMODULE_SPEED	Geschwindigkeit unabhängig vom Typ der OeV-Anfrage

8 LAUFENDE ERWEITERUNGEN UND KORREKTUREN

8.1 Verbesserungen aus vs | plus 9.1.0 (Patch 6)

Betrifft Verbesserungen, die in Version 9.1.0 nach dem Release der ersten Version 10.0.0 unternommen und anschliessend übertragen wurden.

	Erweiterungen und Fehlerbehebungen der vs | plus Bedarfssteuerung (Patch 6)
U 10474 / U 10520	Ab Version 9.1.0 [2][1] (Patch 6) wurden folgende Probleme der Bedarfssteuerung behoben: - Korrektur der Mindestzeitüberwachung bei fehlender Unterscheidung des Steuergeräts zwischen Standard- und Nicht-Standard-Frei-/Sperrzeiten. - Behebung eines Wertüberlaufs, der zu einem stark verzögerten Wiedereintritt in den Bedarfsbetrieb führen konnte. - Entfernung eines doppelten Aufrufs der Bedarfssteuerung bei der Laufzeitversorgung neuer Parameter. *Zudem wurde folgende* ***Erweiterung*** *implementiert:* Die Bedarfssteuerung unterstützt nun gestaffelte Beginne und ermöglicht damit Anwendungsfälle wie das Vorstarten von Signalgruppen. Dafür wurde die Mindestzeitüberwachung von Teilsequenzen auf Sequenzebene erweitert.

	Versorgen der Bedarfssteuerungs-Sequenzen anpassen
U 8941 **U 10107**	*Betrifft die Parametrierung und VCB-Exporte mit der vs | worksuite:* Ab Version 9.1.0 [2][1] (Patch 6) werden bei der Versorgung der **vs | plus** Parameter in der VCB hinterlegte Sequenzen für die Bedarfssteuerung ignoriert, wenn die Bedarfssteuerung in keiner Programmdefinition aktiviert ist. Zuvor führte dies zu einem Versorgungsfehler.

	Ausnahmebehandlung bei open vs | plus Skriptfehler
B 10368	Das Verarbeiten eines open vs | plus Skripts mit erst zur Laufzeit vom Interpreter erkannten Fehlern, führte zu einer Ausnahme, nachdem der Interpreter beendet und eine Warnungsmeldung Nr. 50 ausgegeben wurde. Ab Version 9.1.0 [2][1] (Patch 6) findet ein anschliessender Zugriff auf den Interpreter nicht mehr statt.

8.2 Verbesserungen in Version 10.0.0

Betrifft Verbesserungen die seit dem Release der ersten Version 10.0.0 eingeflossen sind.

B 10599	**Parallele OeV-Abforderungen löschen bestehende Anmeldungen**
	Bei Meldepunkten des Typs «parallel» für die OeV-Abforderung wurde nicht geprüft, ob tatsächlich eine Anforderung des zugeordneten Detektors vorlag. In der Folge wurde die Meldestrecke laufend abgemeldet, sodass bereits vorhandene Anforderungen wieder gelöscht wurden. Ab Version **10.0.0[0][1]** ist dieses Fehlverhalten korrigiert.

B 10601 / B10654	**Abforderung durch open vs | plus OeV-Telegram funktioniert nicht**
	OeV-Telegramme aus open vs | plus wurden in bestimmten Fällen nicht korrekt verarbeitet, da sie vor der Verarbeitung der Meldestrecken aus dem Speicher entfernt wurden. Ab Version **10.0.0[0][1]** werden die Telegramme korrekt zwischengespeichert und beim Aktualisieren der weiteren OeV-Eingänge mit den übrigen Anforderungen zusammengeführt. Dadurch ist die Verarbeitung von OeV-Telegrammen aus open vs | plus wieder sichergestellt.

11/09/2026	**Ausgabe der internen vs | plus Version anpassen**
	Ab Version **10.0.0[0][1]** wurde die interne Versionierung in vs | plus vom Format N.N.N (Major, Minor, Patch) wieder auf das früher verwendete Format umgestellt (N.N.N[N][N]). Zusätzlich werden neue Teil-Nummern in der allgemeinen Schnittstelle definiert: - VSP_INTERFACE_VERSION - VSP_MAJOR_VERSION - VSP_MINOR_VERSION - VSP_FEATURE_PATCH_VERSION - VSP_PATCH_VERSION Die Ausgabe ist wie folgt: **Interface.Major.Minor[FeaturePatch][Patch]**

U 10530	**Linien und Routen Versorgung mit vs | studio abgleichen**
	Ab **vs | studio 3.2** ist das Setzen von «Linien und Routen» für OeV-Versorgungen obligatorisch. Dadurch ist für den Benutzer klar ersichtlich, dass das OeV-Modul mit Linien- und Routenverfolgung arbeitet. Die Vollständigkeit der Daten wird bereits bei der Parametrierung geprüft. Ab **vs | plus 10.0.0[0][1]** wird zudem keine Fehlermeldung Nr. 13 mehr ausgegeben, wenn eine Versorgung ohne Linien- und Routendaten empfangen wird. In diesem Fall werden die fehlenden Angaben als «alle Linien – alle Routen» interpretiert.

U 9832	**Verzögerung Abmelder, Entprellzeit serielle Telegramme berücksichtigen**
	Das Verhalten von Entprell- und Verzögerungszeiten im OeV-Modul wurde überprüft. Die Entprellzeit für serielle OeV-Telegramme wird nicht verwendet. Die Verzögerungszeit für serielle Anmeldetelegramme entspricht weiterhin dem Verhalten aus Version 9.1.0. Eine Anpassung der Verzögerungszeit für Abmeldungen wird vorerst nicht umgesetzt, da der Nutzen den erforderlichen Implementierungsaufwand nicht rechtfertigt. Die Vereinheitlichung erfolgt zu einem späteren Zeitpunkt im Rahmen der Integration des Standard-OeV in das OeV-Modul.

B 10633	**Prüfung der versorgten OeV-Speicher Meldestrecken entschärfen**
	In bestimmten Konstellationen wurde fälschlicherweise die Versorgungsfehlermeldung Nr. 13 ausgegeben, wenn die Anzahl der Verkehrsstrom-Meldestrecken von der Anzahl der Meldepunkt-Meldestrecken abwich. Dabei wurde nicht berücksichtigt, dass Verkehrsstrom-Meldestrecken ohne zugeordnete Meldepunkte zulässig sind und von vs | plus unabhängig vom OeV-Modul verarbeitet werden können. Ab Version **10.0.0[0][1]** wurde die Versorgungsprüfung entsprechend angepasst, sodass nur noch tatsächlich unzulässige Konfigurationen als Fehler erkannt werden.

Impressum

Dateiname	vsplus Version 10 Release Notes DE.docx
Speicherdatum	01.10.2026 14:20
Letzte Änderung	Cyrill Jucker
Qualitätssicherung	SQS-zertifiziertes Qualitätssystem nach ISO 9001 (Reg.Nr. 34856)
© Copyright	vs | verkehrssysteme ag