

# Simulator Exchanging Protokoll

## Anforderungen an eine offene UDP Schnittstelle

### Inhaltsverzeichnis

1	Hintergrund.....	3
2	Allgemeines.....	3
3	Systemschema.....	4
4	Anforderungen.....	5
4.1	Individuelle Beschreibung der Telegramme.....	5
4.2	Telegramme mit Prozesswerten.....	8
4.3	Telegramme mit Messagedaten.....	9
4.4	Datentypen.....	10
4.5	Digitale Prozesswerte.....	12
4.6	Analoge Prozesswerte.....	14
4.7	Messages.....	16
5	Beispiele.....	18
5.1	PZB-Leuchtmelder mit Sifa, Geschwindigkeit und Bremse.....	18
6	Glossar.....	19

# Änderungsjournal

Datum	Bearbeiter	Kommentar	Version
21.07.18	Jens Eggert	First issue.	0.2
12.08.18	Jens Eggert	Modified order of requirements.	0.3
28.08.18	Jens Eggert	Changed XML to JSON.	0.4
13.08.18	Jens Eggert	Changed data format concept.	0.5
11.11.18	Jens Eggert	Added process values.	0.6
15.11.18	Jens Eggert	Removed "size" from example. Added content to RequirementID_2060_p. Added process value.	0.7
01.12.18	Jens Eggert	Changed data types to small letters. Removed quotation marks from port.	0.8
13.02.19	Jens Eggert	Added ENUM for door system	0.9
05.07.19	Uwe Klein	Added process values. Changed data type <bcd> in <uint4>. Rename <IndSimBreak> in <IndSimRun>.	0.91
10.08.19	Uwe Klein	Added process values.	0.92
29.08.19	Uwe Klein	Added process values.	0.93
09.04.20	Uwe Klein	Added process values. Adjusted separator for table elements in strings. Added content to RequirementID_1211_p.	0.94
07.06.20	Jens Eggert	Refined Json "size" and "length". Redesigned InfDoorStatus. Refined InfSimTime. Differentiation between message with flexible and fixed length in 3030_p, 3040_i and 3050_P. Added chapter EbuLa. Added RequirementID_1171_p (string defaults values).	0.95

## 1 Hintergrund

Der Zeit gibt es verschiedene Ansätze zur Kommunikation von Simulatoren mit externen Programmen oder Hardware. Die Ansätze gehen über eine Kommunikation via TCP/IP, Einbindung eine DLL auf Simulatorseite, Einbindung eine DLL auf Zielanwendungsseite, Parsen von Prozessen und der Verwendung von OLE, das eigentlich für MS-Office-Anwendungen entwickelt wurde.

Die Ansätze haben dabei verschiedene Nachteile. Sie erzeugen viel Overhead, haben keine einheitlichen Telegramme, sind nicht ausreichend dokumentiert oder werden vom Simulatorhersteller nicht offiziell unterstützt. Dieses Dokument ist der Versuch eine Einheitliche Schnittstelle zu definieren, die von möglichst vielen Simulatorherstellern als ein einfach zu implementierender Standard angesehen wird und zu Anwendung kommt..

## 2 Allgemeines

Die Schnittstelle soll „Simulator Exchanging Protokoll“ -SEP- genannt werden. Sie soll einfach zu Interpretieren sein, keinen Verbindungsaufbau erfordern, möglichst wenig Overhead haben, ohne der Verwendung spezieller Bibliotheken zu implementieren sein und von PC-externen, wie internen Anwendungen verwendet werden können.

Das hier beschriebene Protokoll entspricht in Teilen denen, die in Eisenbahnfahrzeugen verwendeten verwendet werden. Unter anderem in den Bussystemen CAN, MVB oder Ethernet.

Konkrete Benennungen von Parametern in diesem Dokument beziehen sich auf den Bahnbereich. Benennungen für andere Anwendungen, wie z.B. Flug- oder Bussimulatoren können anderweitig gewählt werden.

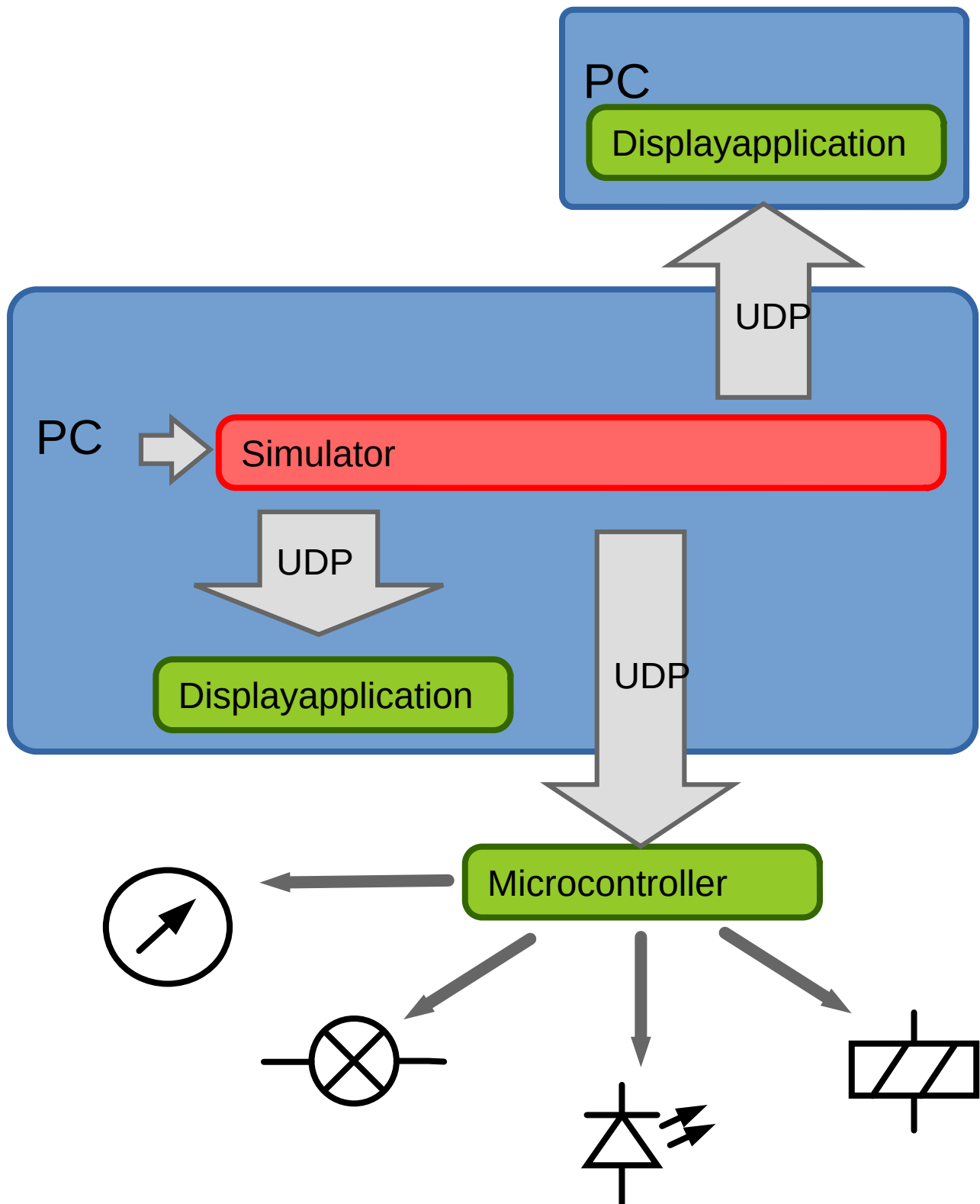
Im folgenden werden Anforderungen aufgelistet. Diese haben ID-Nummern und unterscheiden sich zwischen Pflichtanforderungen \_p, optionalen Anforderungen \_o und informativen Anforderungen \_i.

Hersteller sind dazu angehalten, mindestens alle Pflichtanforderungen zu erfüllen, um Anwendungssoftware kompatibel zu halten.

Im Protokoll wird zwischen Prozesswerten und Messagedaten unterschieden. Prozesswerte bilden den Zustand von Leuchtmeldern, akustischen Signalgebern, Anzeigen von Analogwerten und Textmeldungen aus Textkonserven ab.

Messagedaten enthalten Texte, die für die Anzeige von Fahrplänen, Fahrdienstleiter-Befehlen oder Fahrtsauswertungen verwendet werden können.

### 3 Systemschema



## 4 Anforderungen

### 4.1 Individuelle Beschreibung der Telegramme

#### RequirementID\_1010\_p

Die Beschreibung eines Telegrammaufbaus erfolgt im JSON-Format.

Die JSON-Datei wird „Telegram describer file“ (TDF) bezeichnet und soll die Dateiendung \*.tdf haben.

#### RequirementID\_1020\_p

Im TDF muss für jedes Telegramm ein UDP-Port definiert werden.

#### RequirementID\_1030\_p

Im TDF kann für jedes Telegramm eine IP-Adressen definiert werden.

Das fehlen der Angabe einer IP-Adresse impliziert die Verwendung des localhost.

#### RequirementID\_1040\_p

Das Format eines analogen Prozesswertes wird im TDF definiert.

Es muss die Länge des Wertes angegeben werden.

#### RequirementID\_1050\_p

Für analoge Prozesswerte muss ein Multiplikator im TDF definiert werden können.

#### RequirementID\_1060\_p

Die Anzahl aller verwendeten Bits eines Telegrams muss ein Vielfaches von acht sein.

#### RequirementID\_1070\_p

In der JSON-Struktur werden die Telegramme innerhalb von

```
"telegrams": [{ ... }, { ... }, { ... }]
```

als Array nacheinander notiert.

#### RequirementID\_1080\_p

In der JSON-Struktur wird eine IP-Adresse mit

```
"ip": ...
```

innerhalb eines Telegramms beschrieben.

#### RequirementID\_1090\_p

In der JSON-Struktur wird ein UDP-Port mit

```
"port": 1000
```

innerhalb eines Telegramms beschrieben.

## RequirementID\_1100\_p

In der JSON-Struktur das Format eines Telegramms innerhalb von

```
"format": [{ ... }, { ... }, { ... }]
```

innerhalb eines Telegramms beschrieben.

## RequirementID\_1112\_o

In der JSON-Struktur kann eine Hysterese mit

```
"hysteresis": 10
```

innerhalb eines Formats angegeben werden.

## RequirementID\_1130\_p

In der JSON-Struktur wird die Dimensionierung eines Element mit der Wahl eines Datentyps

```
"type": "uint16" oder z.B. "type": "digital"
```

innerhalb eines Formats beschrieben.

*Siehe Tabelle 1 für weitere möglichen Datentypen.*

## RequirementID\_1140\_p

In der JSON-Struktur wird der Name eines analogen Elements mit

```
"name": " ... "
```

beschrieben und muss innerhalb des analogen Elements angegeben werden.

## RequirementID\_1150\_p

In der JSON-Struktur wird ein Multiplikator eines analogen Elements mit

```
"factor": ...
```

beschrieben und muss innerhalb des analogen Elements angegeben werden.

## RequirementID\_1160\_p

In der JSON-Struktur wird die Größe eines analogen Elements in Byte mit

```
"dimension": ...
```

beschrieben und muss innerhalb des analogen Elements angegeben werden.

## RequirementID\_1170\_p

In der JSON-Struktur wird der Defaultwert eines analogen Elements mit

```
"default": ...
```

beschrieben und innerhalb des analogen Elements oder String angegeben.

## RequirementID\_1171\_p

Defaultwerte für Strings müssen um Base64-Format angegeben werden.

### **RequirementID\_1180\_p**

In der JSON-Struktur wird ein Timer mit

```
"type": "timer"
```

innerhalb eines Formats beschrieben.

### **RequirementID\_1190\_p**

In der JSON-Struktur wird die Größe eines Timers mit

```
"size": ...
```

in Bits innerhalb eines Formats eines Timers angegeben.

### **RequirementID\_1200\_p**

In der JSON-Struktur kann eine Zykluszeit in Millisekunden mit

```
"cycle": 250
```

innerhalb eines Formats beschrieben werden.

### **RequirementID\_1210\_p**

In der JSON-Struktur wird der Defaultwert eines Timers mit

```
"default": ...
```

innerhalb eines Formats eines Timers angegeben.

### **RequirementID\_1211\_p**

In der JSON-Struktur wird die Länge eines Strings mit

```
"length": ...
```

in Bytes innerhalb eines Formats eines Strings angegeben.

### **RequirementID\_1220\_o**

Der Name eines Prozesswertes oder von Messagedaten sollte, sofern enthalten, der Namensliste dieses Dokuments entnommen werden. Siehe Tabellen.

### **RequirementID\_1230\_o**

Die Namen von Prozesswerten und Messagedaten sollte, sofern nicht in diesem Dokument enthalten, auf Englisch gewählt werden.

### **RequirementID\_1240\_o**

Für Prozesswerte können Defaultwerte im TDF definiert werden. Diese sollen nicht durch einen Multiplikator verändert werden.

### **RequirementID\_1250\_o**

Für jedes Telegramm kann ein Zähler als Lebenszeichen definiert werden. Für diesen ist Größe in Byte anzugeben.

### **RequirementID\_1260\_i**

Im TDF können beliebig viele Telegramme definiert werden.

### **RequirementID\_1270\_i**

Im TDF werden für jedes Telegramm entweder Prozesswerte oder Messagedaten definiert.

## **4.2 Telegramme mit Prozesswerten**

### **RequirementID\_2010\_p**

Prozesswerte müssen, der Reihenfolge aus dem TDF entsprechend, im Telegramm übertragen werden.

### **RequirementID\_2020\_p**

Prozesswerte müssen unmittelbar aufeinander folgen.

### **RequirementID\_2030\_p**

Für analoge Prozesswerte sind Fließkommazahlen nicht möglich.

*Zur Darstellung von Nachkommastellen in einem Analogwert kann der Multiplikator aus dem TDF verwendet werden. Z.B. bewirkt die Multiplikation mit 100 die Darstellung von zwei Nachkommastellen.*

### **RequirementID\_2040\_p**

Prozesswerte des Typs Indicator bestehen aus 4 Bit.

Bit 0: Das Element ist aktiv.

### **RequirementID\_2050\_p**

Digitale Prozesswerte bestehen aus einem Bit.

### **RequirementID\_2060\_p**

Telegramme sollen ereignisbezogen übertragen werden, wenn Zykluszeit definiert ist. Ist eine Hysterese angegeben, muss diese für ein Übertragungsereignis berücksichtigt werden.

### **RequirementID\_2070\_p**

Telegramme müssen zyklisch zu übertragen werden, wenn eine Zykluszeit im TDF definiert ist.

### **RequirementID\_2080\_p**

Ein Zähler ist ein Wert, der zu jeder Übertragung inkrementiert werden soll. Werden Zähler nicht unterstützt, muss im Telegrammen für den Zähler der Defaultwert übertragen werden.

### **RequirementID\_2090\_p**

Der Überlauf eines Zählers führt zum Neubeginn bei Null.



### **RequirementID\_2091\_p**

Ist im TDF für einen Element eine Hysterese definiert und ist keine Zykluszeit definiert, soll ein Telegramm dann übertragen werden, wenn sich der Betrag des Elements um mehr als den Betrag der Hysterese ändert.

### **RequirementID\_2100\_o**

Für Prozesswerte des Typs Indicator kann, z.B. für Leuchtmelder, ein Blinken, schnelles Blinken und inverses Blinken definiert werden.

Bit 0: Das Element ist aktiv.

Bit 1: Das Element schaltet mit 1Hz.

Bit 2: Das Element schaltet mit 2Hz.

Bit 3: Das Element arbeitet invertiert.

Wird diese Anforderung unterstützt, muss Bit 0 auch bei einem Blinken immer gesetzt sein.

## **4.3 Telegramme mit Messagedaten**

### **RequirementID\_3010\_p**

Die enthaltene Information soll als Text im UTF-8-Format gestaltet werden.

### **RequirementID\_3020\_p**

Tabellenspalten werden durch Strichpunkte ';' separiert. Tabellenzeilen werden durch Zeilenumbrüche '\n' separiert.

### **RequirementID\_3030\_p**

Ist der enthaltene Text kürzer als das Telegram, muss das Ende der Message mit dem Nullterminator '\0' markiert werden.

### **RequirementID\_3040\_i**

Wird keine Länge angegeben, kann diese flexibel sein und sich nach dem Inhalt richten.

### **RequirementID\_3050\_p**

Wird eine Länge angegeben, muss das Messagedatum immer diese Länge haben.

## 4.4 Datentypen

Im folgenden werden Datentypen beschrieben, die dem Protokoll mindestens zu Verfügung stehen.

Name	Größe [Bit]	Anwendung
digital	1	Einfache digitale Informationen.
indicator	4	Leuchtmelder und Akkustische Geber.
uint4	4	Analoge Informationen mit Werten von 0 bis 16
uint8	8	Analoge Informationen mit Werten von 0 bis 255
uint16	16	Analoge Informationen mit Werten von 0 bis 65.535
uint32	32	Analoge Informationen mit Werten von 0 bis 4.294.967.295
int8	8	Analoge Informationen mit Werten von -128 bis 127
int16	16	Analoge Informationen mit Werten von -32.768 bis 32.767
int32	32	Analoge Informationen mit Werten von -2.147.483.648 bis 2.147.483.647
string		Zeichenkette

Tabelle 1

### RequirementID\_401\_p

Der Datentyp Digital besteht aus einem Bit.

*Wichtig: Er ist nicht wie in den meisten Programmiersprachen als ein Boolean zu betrachten, das aus 8 Bit besteht.*

### RequirementID\_402\_p

Der Datentyp Indicator besteht aus vier Bit. Er bildet eine komplexe Information über den Zustand eines Leuchtmelders oder Tongebers ab.

Bit 0: Das Element ist aktiv.	1
Bit 1: Das Element toggelt mit 1Hz.	2
Bit 2: Das Element toggelt mit 2Hz.	4
Bit 3: Das Element arbeitet invertiert.	8

### RequirementID\_403\_p

Der Datentyp uint4 besteht aus 4 Bit und hat kein Vorzeichen.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### RequirementID\_404\_p

Der Datentyp uint8 besteht aus 8 Bit und hat kein Vorzeichen.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### **RequirementID\_405\_p**

Der Datentyp uint16 besteht aus 16 Bit und hat kein Vorzeichen.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### **RequirementID\_406\_p**

Der Datentyp uint32 besteht aus 32 Bit und hat kein Vorzeichen.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### **RequirementID\_407\_p**

Der Datentyp int8 besteht aus 8 Bit und ist vorzeichenbehaftet.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### **RequirementID\_408\_p**

Der Datentyp int16 besteht aus 16 Bit und ist vorzeichenbehaftet.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### **RequirementID\_409\_p**

Der Datentyp int32 besteht aus 32 Bit und ist vorzeichenbehaftet.

Das höchstwertige Byte steht am Ende. (*Big-Endian*)

### **RequirementID\_4010\_p**

Der Datentyp String kann mit einer Länge von 1 bis 65.535 definiert werden.

*Dabei entspräche die Länge von 1, einem einzelnen Zeichen.*

*65,507 ist die maximale Länge eines UDP-Packets. Diese Länge kann also nur erreicht werden, wenn kein anderer Prozesswert im Telegramm enthalten ist. Der Nullterminator ist in der Länge enthalten.* Verfügbare Prozesswerte

## 4.5 Digitale Prozesswerte

Die Liste der folgenden Prozesswerte hat noch keinen Anspruch auf Vollständigkeit.

### 4.5.1 LZB/PZB

Benennung	Beschreibung
IndLzbBef40	PZB-Leuchtmelder „Befehl 40“
IndLzb1000	PZB-Leuchtmelder „1000Hz“
IndLzb500	PZB-Leuchtmelder „500Hz“
IndLzbO	PZB-Leuchtmelder „85“
IndLzbM	PZB-Leuchtmelder „70“
IndLzbU	PZB-Leuchtmelder „55“
IndLzbH	LZB-Leuchtmelder „H“
IndLzbG	LZB-Leuchtmelder „G“
IndLzbE40	LZB-Leuchtmelder „E40“
IndLzbEl	LZB-Leuchtmelder „El“
IndLzbEnd	LZB-Leuchtmelder „Ende“
IndLzbV40	LZB-Leuchtmelder „V40“
IndLzbB	LZB-Leuchtmelder „B“
IndLzbS	LZB-Leuchtmelder „S“
IndLzbUe	LZB-Leuchtmelder „Ü“
IndLzbStoe	LZB-Leuchtmelder „Stör“
IndLzbHupe	Akustischer Signalgeber PZB-Hupe
IndLzbSchn	Akustischer Signalgeber LZB-Schnarre
IndLzbFb	LZB/PZB-Zwangsbremmung
IndLzbTl	LZB/PZB-Traktionssperre
IndLzbShow	LZB-Führungsgrößen anzeigen.
InfLzbSystem	Verwendetes PZB/LZB System (8Bit) <ol style="list-style-type: none"> <li>1. I60</li> <li>2. I60 mit ER2</li> <li>3. I60 mit ER24 und PZB90</li> <li>4. I60R</li> <li>5. I60R mit PZB90</li> <li>6. I80</li> <li>7. I80 mit PZB90</li> <li>8. PZ80</li> <li>9. PZ80R</li> <li>10. PZ80R mit PZB90</li> </ol>
InfPzbZa	PZB Zugart (4Bit) <ol style="list-style-type: none"> <li>1. U</li> <li>2. M</li> <li>3. O</li> </ol>

Tabelle 2

### 4.5.2 Maschinentechnisch

Benennung	Beschreibung
IndMtMsO	Leuchtmelder Hauptschalter aus
IndMtHD	Leuchtmelder hohe Abbremsung
IndMtEB	Leuchtmelder Elektrische Bremse
IndMtHvtl	Leuchtmelder Zugsammelschiene
IndMtEm	Leuchtmelder Notbremsung
IndWhSp	Leuchtmelder Schleudern
IndSlipp	Leuchtmelder Gleiten
IndDbc	AFB aktiv bzw. eingeschaltet
tractionType	Traktionsart 1. Elektrisch 2. Diesel 3. Dampf
IndDss	Richtungswahlschalter: 1: Vorwärts, 2: Rückwärts
IndPanto	Stellung Stromabnehmer (Pantograph) 1. gehoben 2. wird gehoben 3. wird gesenkt 4. gesenkt

Tabelle 3

### 4.5.3 Sifa

Benennung	Beschreibung
IndDsd	Leuchtmelder „Sifa“
IndDsdAcu	Akustischer Signalgeber „Sifa“
IndDsdFb	Sifa Zwangsbremse
IndDsdTl	Sifa Traktionssperre

Tabelle 4

#### 4.5.4 Türsystem

Benennung	Beschreibung
IndDoorLOpen	Türen Links offen
IndDoorROpen	Türen Rechts offen
IndDoorSat	Leuchtmelder Türsystem SAT
IndDoorTav	Leuchtmelder Türsystem TAV
IndDoorTl	Türsystem Traktionssperre
InfDoorSystem	Verwendetes Türsystem (8 Bit) 1. undefiniert 2. SAT 3. TB0 4. TAV 5. SST
InfDoorStatusL InfDoorStatusR	Status des Türsystems bezogen auf eine Seite (4 Bit) Bit 1: Freigabe Bit 2: Offen Bit 3: Schließvorgang Bit 4: Fahrgastwechsel

Tabelle 5

#### 4.5.5 Sonstiges

Benennung	Beschreibung
IndSimRun	Der Simulator ist aktiv
IndSimFf	Der Simulator wird mit Zeitraffer ausgeführt
IndBlanc	Leeres Feld. Ein Element, das immer Null ist.

Tabelle 6

### 4.6 Analoge Prozesswerte

#### 4.6.1 PZB/LZB

Benennung	Beschreibung	Einheit
LzbSpeedPermitted	LZB Soll-Geschwindigkeit	km/h
LzbSpeedTarget	LZB Ziel-Geschwindigkeit	km/h
LzbTargetDistance	LZB Ziel-Entfernung	m
LzbSpeedVue	LZB/PZB-Überwachungskurve	km/h
PzbSpeedTarget	PZB Ziel-Geschwindigkeit	km/h

Tabelle 7

## 4.6.2

### 4.6.3 Maschinentechnisch

Benennung	Beschreibung	Einheit
speed	Ist-Geschwindigkeit in km/h	km/h
DbcSpeedSet	Sollgeschwindigkeit aus AFB	km/h
mainBrakePipe	Hauptluftleitung	bar
brakeCylinderPressure	Bremszylinderdruck	bar
mainAirReservoir	Hauptluftbehälter	bar
timeContainerPressure	Zeitbehälter	bar
brakingForceAbs	Bremskraft absolut	kN
brakingForceRel	Bremskraft relativ	%
tractiveForceAbs	Zugkraft absolut	kN
tractiveForceRel	Zugkraft relativ	%
tractionMode	Fahrstufe	-
maxSpeedEng	Höchstgeschwindigkeit Fahrzeug	km/h
maxSpeedTrain	Höchstgeschwindigkeit Zug	km/h
ContactLineVoltage	Fahrleitungsspannung	V
ContactLineCurrent	Oberstrom	A

Tabelle 8

## 4.6.4

### 4.6.5 Sonstiges

Benennung	Beschreibung
telegramIndicator	In Verbindung mit Defaultwert zur Identifikation eines Telegramms
InfSimTime	Simulatorzeit in Sekunden (32 Bit Unixzeit <u>ohne</u> Vorzeichen)
InfSimAbsPos	Absolute Position auf der Strecke (Hektometer) [m]
InfSimRelPos	Relative Position auf der Strecke zur Gesamtstrecke [m]
InfSimDis	Gefahrene Wegstrecke [m]

## 4.7 Messages

Benennung	Beschreibung
textTimeTable	Fahrplan (Generisch)
textEbuLa	EbuLa-Tabelle (Gesondert spezifiziert)
textMessageDmi	Textnachricht zur Anzeige im DMI nach ERA-Layout
textDispVerbal	Mündlicher Befehl vom Fahrdienstleiter
textDispWritten	Schriftlicher Befehl vom Fahrdienstleiter

Tabelle 9

### 4.7.1 EBUla

Die Tabelle für das EbuLa wird als CSV-String übertragen. Jede Zeile wird in folgender Struktur beschrieben:

```
relativePosition;[limit];[limitGnt];[limitNotSignaled];
[labelPosition];[featOppositeTrack];[tracks];[tunnel];
[featMainTrack];[text];[saw];[arrive];[depart] ↵
```

Mit Ausnahme von `relativePosition` sind alle Elemente optional.

Benennung	Beschreibung
relativePosition	Position bezogen auf den gefahrenen Weg.
Limit	Geschwindigkeitsänderung für nicht Neigetechnikzüge
LimitGnt	Geschwindigkeitsänderung für Neigetechnikzüge.
limitNotSignaled	"1", wenn die Geschwindigkeitsänderung nicht durch Signale angezeigt wird.
labelPosition	Die Position, bezogen auf die Strecke, die in der Tabelle angezeigt werden soll.
featOppositeTrack	Piktogramme, die zum Gegengleis angezeigt werden sollen. Siehe Tabelle 11.
tracks	" " oder "1": Einleisig "  " oder "2": Zweigleisig. "   " oder "3": Fahren im Gegengleis
tunnel	"⌒" oder "1": Tunnelanfang "  " oder "2": Tunnelinnere. "⌒" oder "3": Tunnelende
featMainTrack	Piktogramme, die zum Regelgleis angezeigt werden sollen.
text	Freier Text der Betriebsstelle
saw	"1", wenn eine Sägezahnlinie angezeigt werden soll. "2", wenn zwei Sägezahnlinien angezeigt werden sollen.
arrive	Ankunftszeit im Format hh:mm.s.
depart	Abfahrtszeit im Format hh:mm.s.

Tabelle 10



## featOppositeTrack & featMainTrack

Nachfolgend werden Texte aufgelistet, die die möglichen Piktogramme repräsentieren. Optional kann eine Nummerierung nach Ril 408.0341A01 oder Text mit Unicodezeichen verwendet werden, dass dem Piktogramm ähnelt und so den Text lesbarer macht.
















Nummer	Text	Piktogramm
y1	¥	¥
a1	▽	▽
b1	┌	┌
c1	LZB	
c2	LZB Ende	
d1	GNT	
d2	GNT Ende	
f1	=	
g1	Heizverbot	
g2	Heizverbot Ende	
e1	⌒	
e2	⌒'	
f5	⓪	
f6	⓪'	
f4	◈	
f2	◈Erw	
f3	◈	
h1	◇	

Tabelle 11

## 5 Beispiele

### 5.1 PZB-Leuchtmelder mit Sifa, Geschwindigkeit und Bremse

#### TelegramDescriber.tdf

```
{
  "telegrams": [
    {
      "IP": "192.168.0.10"
      "port": 1001
      "format": [
        { "type": "indicator", "name": "IndBlanc"}, // 4 leere Bits
        { "type": "indicator", "name": "IndLzbBef40"}, // Lm Bef.40
        { "type": "indicator", "name": "IndLzb1000"}, // Lm 1000Hz
        { "type": "indicator", "name": "IndLzb500"}, // Lm 500Hz
        { "type": "indicator", "name": "IndLzbO"}, // Lm 85
        { "type": "indicator", "name": "IndLzbM"}, // Lm 70
        { "type": "indicator", "name": "IndLzbU"}, // Lm 55
        { "type": "indicator", "name": "IndDsd"} // Lm Sifa
      ]
    },
    {
      "IP": "192.168.0.10"
      "port": 1002
      "format": [
        { "type": "uint16", "name": "speed", "factor": 10}, // Vist
        { "type": "uint16", "name": "mainBrakePipe", "factor": 100}, // HLL
        { "type": "uint16", "name": "brakeCylinderPressure", "factor": 100}, // BRZ
        { "type": "uint16", "name": "mainAirReservoir", "factor": 100, "default": 10000} // HLB
      ]
    }
  ]
}
```

#### Telegramm 1 an 192.168.0.10 auf Port 1001 (4 Byte)

0000	0000	0001	0000	0011	0000	0000	0000
	Bef. 40	1000Hz	500Hz	85	70	55	Sifa

#### 1000Hz-Überwachung

0000	0000	0000	0001	0011	1011	0000	0000
	Bef. 40	1000Hz	500Hz	85	70	55	Sifa

#### 500Hz-Überwachung restriktiv

#### Telegramm 2 an 192.168.0.10 auf Port 1002 (8 Byte)

00000001	01001010	00000001	11001000	00000000	00010100	00000011	11101000
Geschwindigkeit		Hauptluftleitung		Bremszylinder		Hauptluftbehälter	

33km/h, 4,56bar HLL, 0,2bar Bremszylinder, 10bar Hauptluftbehälter

## 6 Glossar

Abkürzung	Bedeutung
ASCII	American Standard Code for Information Interchange Eine Codierung von Satzzeichen und Alphabet.
CAN	Controller Area Network. Ein Bussystem, das in Industrie Fahrzeugen eingesetzt wird.
DLL	Dynamic Link Library Eine Programmbibliothek mit einer Sammlung von Unterprogrammen.
EBuLa	Elektronischer Buchfahrplan und Langsamfahrstellen
Ethernet	Ein Bussystem, das zur Vernetzung von Computern verwendet wird.
IP	Internetprotokoll Ein Protokoll zur Übertragung durch Computernetze.
IP-Adresse	Die Adresse eines Teilnehmers eines IP-Computernetz.
JSON	JavaScript Object Notation Eine Sprache zur Darstellung und zum Austausch strukturierter Daten.
localhost	Die eigene Adresse eines Teilnehmers eines IP-Computernetz.
MVB	Multifunction Vehicle Bus Ein Bussystem, auf Schienenfahrzeugen eingesetzt wird.
OLE	Object Linking and Embedding. Ein Objektsystem zur Erstellung von Verbunddokumenten.
Port	Ordnet Daten innerhalb eines IP-Computernetz einer bestimmten Anwendung auf dem adressierten Computer zu.
RequirementID	Die Nummerierung der Anforderungen in diesem Dokument.
TCP	Transmission Control Protocol Ein auf IP basierendes, verbindungsbehaftetes Datenprotokoll.
UDP	Transmission Control Protocol Ein auf IP basierendes, verbindungsloses Datenprotokoll.
Ril	Konzernrichtlinien der Deutschen Bahn AG

Tabelle 10