

Kernsystem

Verkehrsrechnerzentralen

SW-Einheit KExDaV

V-Modell-Dokumentation

Ersteller:

Kappich
Systemberatung

integrativ und unabhängig
Kompetenz in System- und Verkehrstechnik

Autor:

Dipl.-Ing. C. Westermann
Dipl.-Inform. R. Schmitz
J. Haas

Version: 1.0
Stand 31.05.2011
Status: Akzeptiert
PID: LBNW16.023-KExDaV-1.0
Submodell: ----
Dokument: LBNW.16.023-KExDaV.1.0 [VModell-Dokumentation].doc
VS-Einstufung: ----

Projekt ID AG: ----
Projekt ID AN: LBNW.16.023-KExDaV

Kappich Systemberatung

Im Auftrag des Landesbetriebs Straßenbau NRW

1 Allgemeines

Verteilerliste

Entfällt. Dokumentverteilung entsprechend aktuellem Projektverteiler.

Versionsübersicht

Nr.	Datum	Version	Änderungsgrund	Bearbeiter
1	01.03.11	0.1	Ersterstellung	Westermann
2	31.05.11	1.0	Überführung in den Zustand „Akzeptiert“	Westermann

Tabelle 1-1: Versionsübersicht

Änderungsübersicht

Nr.	Version	geändertes Kapitel	Beschreibung der Änderung
1	0.1	alle	Ersterstellung
2	1.0	alle	Überführung in den Zustand „Akzeptiert“

Tabelle 1-2: Änderungsübersicht

Kurzbeschreibung

In diesem Dokument erfolgt die V-Modell Dokumentation zu KExDaV.

Inhalt

1 Allgemeines	2
Verteilerliste	2
Versionsübersicht	2
Änderungsübersicht	2
Kurzbeschreibung	2
Inhalt	3
Abkürzungen	6
Definitionen	6
Verzeichnis der Tabellen	6
Verzeichnis der Abbildungen	6
Referenzierte Dokumente	7
2 Ausgangslage	8
3 Anwenderforderungen an die SW-Einheit "KEx-DaV"	10
4 Technische Anforderungen an die SW-Einheit "KEx-DaV"	12
4.1 Gesamtfunktion des Elements	12
4.1.1 Allgemeine Technische Anforderungen	12
4.1.2 Start und Initialisierung von KExDaV	13
4.1.2.1 Verbindungsaufbau zu den Remote-Datenverteilern	15
4.1.2.2 Abbruch der Datenverteilterverbindung zum Lokal-System	16
4.1.3 Austausch von Onlinedaten	16
4.1.3.1 Konfigurationsobjektauswahl	17
4.1.3.1.1 Ermittlung korrespondierender Konfigurationsobjekte	19
4.1.3.2 Auswahl Onlinedaten	20
4.1.3.2.1 Behandlung bei unterschiedlichen Konfigurationsständen für Attributgruppen oder Aspekte	21
4.1.3.2.2 Behandlung von Referenzen auf Objekte in den Onlinedatensätzen	21
4.1.4 Austausch von Parameterdaten	22
4.1.4.1 Konfigurationsobjektauswahl	22
4.1.4.2 Parameterdaten	22
4.1.4.2.1 Verwaltung der Parameter im Lokal-System (Remote-System nur Lesen)	24
4.1.4.2.2 Verwaltung der Parameter im Lokal-System (Remote-System Lesen und Ändern)	25
4.1.4.2.3 Verwaltung der Parameter im Remote-System (Lokal-System nur Lesen)	25
4.1.4.2.4 Verwaltung der Parameter im Remote-System (Lokal-System Lesen und Ändern)	26
4.1.4.2.5 Verwaltung der Parameter im Lokal- und Remote-System	26
4.1.4.2.6 Verwaltung der Parameter im Lokal- und Remote-System (Austausch nur bei Triggersignal)	27
4.1.4.2.7 Triggerung für den Parameteraustausch	27

4.1.4.2.8	Behandlung bei unterschiedlichen Konfigurationsständen für Parameterattributgruppen oder Aspekte	28
4.1.4.2.9	Behandlung von Referenzen auf Objekte in den Parameterdatensätzen	28
4.1.5	Plugin-Schnittstelle für den Austausch von Datensätzen zu Attributgruppen	28
4.1.6	Austausch von dynamischen Objekten	29
4.1.6.1	Initialisierung	31
4.1.6.2	Laufender Betrieb	32
4.1.6.3	Parameteränderung	33
4.1.6.4	Kopieren eines dynamischen Objektes	33
4.1.6.5	Löschen eines dynamischen Objektes	33
4.1.7	Austausch von dynamischen Mengen	34
4.1.7.1	Anwendungsfall Meldungen	34
4.1.7.2	Anwendungsfall Staus	37
4.1.7.3	Spezifikation Austausch von dynamischen Mengen	38
4.1.7.4	Initialisierung	39
4.1.7.5	Laufender Betrieb	39
4.1.7.6	Parameteränderung	39
4.2	Technische Anforderungen an die Schnittstellen	40
4.2.1	Technische Anforderungen an die Nutzerschnittstelle	40
4.2.1.1	Schnittstelle KExDaV – Starter	40
4.2.2	Technische Anforderungen an andere Schnittstellen	40
4.2.2.1	Schnittstelle KExDaV – Starter	40
4.2.2.2	Schnittstelle KExDaV – Applikation	40
4.3	Qualitätsforderungen	41
4.3.1	Kritikalität	41
4.3.2	Technische Anforderungen der IT-Sicherheit	41
4.3.3	Technische Anforderungen an sonstige Qualitätsmerkmale	41
4.3.4	Technische Anforderungen an die Entwicklungs- und SWPÄ-Umgebung	41
5	SW Architektur der SW-Einheit "KExDaV"	42
5.1	Lösungsvorschläge	42
5.1.1	Kriterien für die Zerlegung der SW-Einheit	42
5.1.2	Skizzierung der gewählten Lösung	43
5.2	Modularisierung/Datenbankentwurf	43
5.2.1	Übersicht der SW-Komponenten, SW-Module, Prozesse und Datenbanken	43
5.2.2	Einzelbeschreibung	44
5.2.2.1	Modul Verwaltung	44
5.2.2.2	Modul Parameter-Ermittlung	44
5.2.2.3	Modul Remote-Verbindung	44
5.2.2.4	Modul Austausch Daten	45
5.2.2.5	Modul Austausch Parameter	45
5.2.2.6	Modul Datenaustausch-Plugin	45
5.2.2.7	Modul Austausch dynamische Objekte	45
5.2.2.8	Modul Austausch dynamische Mengen	45
5.2.3	Dynamisches Ablaufmodell	45

5.2.4	Kritikalität der SW-Komponenten/SW-Module/Prozesse/Datenbanken	46
5.2.5	Sonstige Entwurfsentscheidungen	46
5.3	Schnittstellen	46
5.3.1	Externe Schnittstellen der SW-Einheit	46
5.3.2	Interne Schnittstellen der SW-Einheit	46
5.4	Anforderungszuordnung	48
6	Prüfspezifikation SW-Einheit KExDaV	49
6.1	Anforderungen	49
6.1.1	Einstufung der Funktionseinheit hinsichtlich Kritikalität und IT-Sicherheit	49
6.1.2	Prüfanforderungen	49
6.2	Methoden der Prüfung	49
6.3	Prüfkriterien	50
6.3.1	Abdeckungsgrad	50
6.3.2	Checklisten	50
6.3.3	Endekriterien	50
6.4	Prüffälle	50
6.4.1	Prüffallbeschreibung	51
6.4.1.1	Prüffall 1: Review	51
6.4.1.2	Prüffall 2: TestKExDaV	51
6.4.1.3	Prüffall 3: TestKExDaVObjekt	57
6.4.1.4	Prüffall 4: TestAdjustableTimer	58
6.4.1.5	Prüffall 5: TestBasicKExDaVDataPlugin	58
6.4.1.6	Prüffall 6: TestLowLevelDataPipe	58
6.4.1.7	Prüffall 7: TestKExDaVLocalApplication	58
6.4.1.8	Prüffall 8: TestKExDaVWrappedReferenceValue	58
6.4.1.9	Prüffall 9: TestApplicationStartExit	58
6.4.2	Abdeckungsmatrix	59

Abkürzungen

siehe Dokument "Abkürzungen".

Definitionen

siehe Dokument "Glossar".

Verzeichnis der Tabellen

Tabelle 1-1: Versionsübersicht	2
Tabelle 1-2: Änderungsübersicht	2
Tabelle 5-1: Identifizierung der SW-Komponenten und Module der SW-Einheit KExDaV	44
Tabelle 5-2: Kritikalität der SW-Komponenten/SW-Module/Prozesse/Datenbanken der SW-Einheit KExDaV	46
Tabelle 5-3: Zuordnung der Anforderungen an die SW-Module der SW-Einheit KExDaV	48
Tabelle 6-1: Abdeckungsmatrix	59

Verzeichnis der Abbildungen

Abbildung 2-1: Topologie VRZ Leverkusen	8
Abbildung 3-1: Integration einer neuen Applikation	10
Abbildung 4-1: Kopplung zweier Datenverteilerbasierten Systeme mittels KExDaV	13
Abbildung 4-2: Parameterattributgruppe SpezifikationKExDaV	14
Abbildung 4-3: Austausch von Onlinedaten	16
Abbildung 4-4: Konfigurationsobjektauswahl	17
Abbildung 4-5: Unterschiedliche Konfigurationsstände	19
Abbildung 4-6: Auswahl Onlinedaten	20
Abbildung 4-7: Austausch von Parameterdaten	22
Abbildung 4-8: Strategie Verwaltung Lokal-System (nur Lesen)	24
Abbildung 4-9: Strategie Verwaltung Lokal-System (Lesen und Ändern)	25
Abbildung 4-10: Strategie der Parameter im Lokal- und Remote-System	26
Abbildung 4-11: Attributgruppe TriggerKExDaV	27
Abbildung 4-12: Parameter Austausch von dynamischen Objekten	29
Abbildung 4-13: Ablauf Austausch von dynamischen Objekten	31
Abbildung 4-14: Austausch dynamische Menge Meldungen Variante I	35
Abbildung 4-15: Schema Betriebsmeldungsverwaltung	36
Abbildung 4-16: Austausch dynamische Menge Meldungen Variante II	36
Abbildung 4-17: Austausch dynamische Menge Staus	37
Abbildung 4-18: Parameter Austausch von dynamischen Mengen	38
Abbildung 5-1: Zerlegung der SW-Einheit KExDaV	43

Referenzierte Dokumente

[Afo]	Anwenderforderungen AK VRZ, Dokument "SE-02.00.00.00.00-Afo", aktueller Stand
[SysArc]	Systemarchitektur AK VRZ, Dokument "SE-02.00.00.00.00-SysArc", aktueller Stand
[TAnfGes]	Technische Anforderungen an das Gesamtsystem AK VRZ, Dokument "SE-02.00.00.00.00-TAnf", aktueller Stand
[TAnfDaV]	Technische Anforderungen an den Datenverteiler AK VRZ, Dokument "SE-02.01.00.00.00-TAnf", aktueller Stand

2 Ausgangslage

Im Normalfall werden Datenverteilerbasierte Systeme (z.B. eine Verkehrsrechnerzentrale mit verschiedenen Unterzentralen) über Datenverteiler-Datenverteiler-Verbindungen miteinander gekoppelt.

Abbildung 2-1 skizziert (ohne Anspruch auf Vollständigkeit) einen Teil der aktuellen Topologie zur VRZ Leverkusen (Stand Anfang 2011).

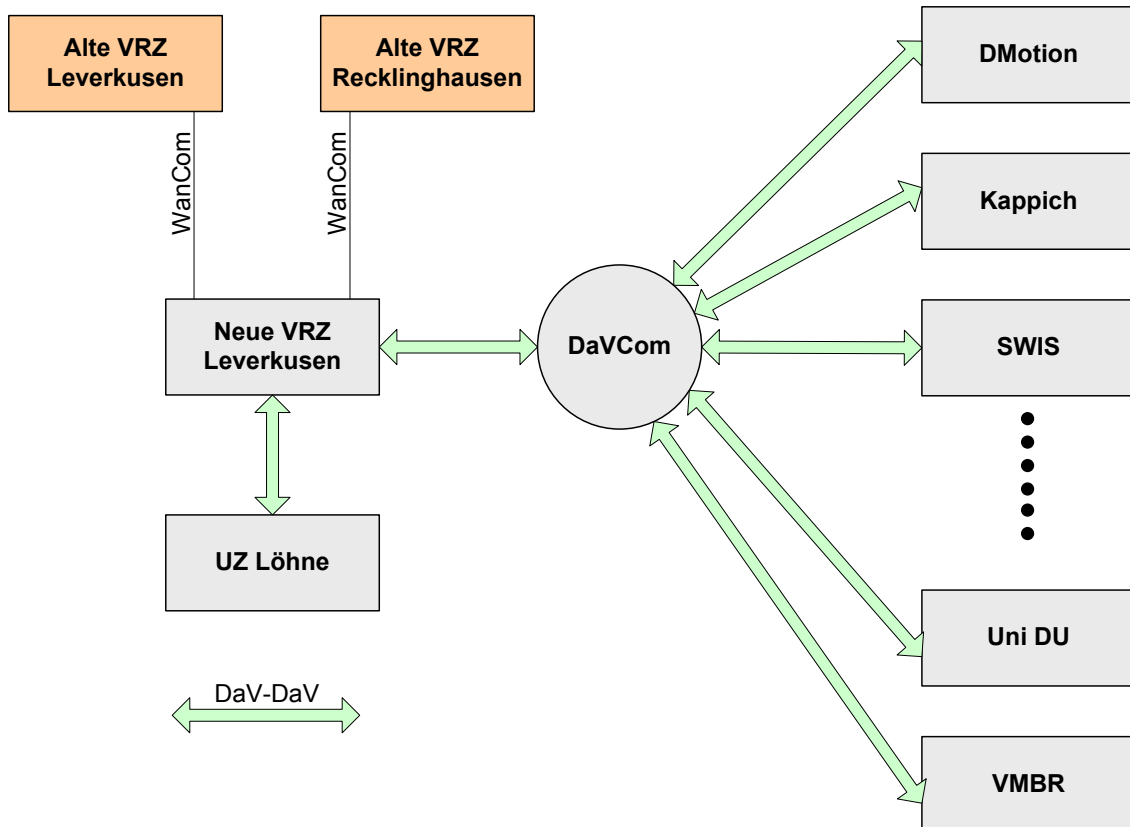


Abbildung 2-1: Topologie VRZ Leverkusen

Das Hauptsystem stellt das Datenverteilersystem VRZ (Neue VRZ Leverkusen) dar. An dieses System sind über WanCom-Verbindungen über die Applikation KExTLS die beiden alten Verkehrsrechnerzentralen Leverkusen und Recklinghausen angeschlossen. Über diese Verbindung werden ausgewählte Daten zwischen der neuen und alten Welt ausgetauscht. Dazu gehören z.B. die Ergebnismeldungen und DE-Fehler zu TLS-Daten sowie die Kurzzeitintervalldaten zu den Fahrstreifen.

Auf dem Hauptsystem laufen z.Z. (Anfang 2011):

- Datenverteiler
- Konfiguration
- Parametrierung (KS)
- Betriebsmeldungsverwaltung (KS)
- Archivsystem (KS)
- KEx-TLS
- Datenaufbereitung LVE (Dambach)
UZ Kaarst, Leverkusen, Ratingen
- ZwischenschichtAq (Dambach)

Vom Hauptsystem gehen zwei Datenverteiler-Datenverteiler-Verbindungen, eine Verbindung zur UZ Löhne und eine zum DaVCom. Auf dem DaVCom-System läuft ein einzelner Datenverteiler als Proxy, über den externe Kommunikationspartner, zum Teil mit eigenen Datenverteilersystemen zum Teil über einzelne Applikationen, angebunden sind.

Die einzelnen Datenverteiler-Systeme sind über die Konfigurationen voneinander abhängig. Das bedeutet, dass es z.B. für den Datenaustausch erforderlich ist, dass die Systeme möglichst gleiche Konfigurationsstände haben. Wenn z.B. zu einem Messquerschnitt MQ 1 (PID `mq1`) Analysewerte ausgetauscht werden sollen (Attributgruppe VerkehrsDatenKurzZeitMq, PID `atg.verkehrsDatenKurzZeitMq`; Aspekt Analyse, PID `asp.analyse`) müssen in den Konfigurationen der Systeme die gleichen Objekte aktiviert sein; sprich in den Konfigurationen muss das gleiche Konfigurationsobjekt zu dem Messquerschnitt z.B. Id 4711, der Attributgruppe VerkehrsDatenKurzZeitMq Id 4712 und dem Aspekt Analyse Id 4713 aktiviert sein.

Eine weitere wichtige Eigenschaft bei über Datenverteiler-Datenverteiler-Verbindungen gekoppelten Systemen ist, dass sich die Systeme so verhalten, als ob nur ein logischer Datenverteiler vorhanden ist. Die Systeme verschmelzen sozusagen zu einem System mit den wichtigen Paradigmen, dass es für eine Datenidentifikation nur eine Quelle bzw. Senke geben kann.

Die aufgeführten Punkte sind im Normalbetrieb von gekoppelten Systemen sehr wichtig und stellen die Integrität der ausgetauschten Daten sicher.

Integration neuer SW

Für die Integration neuer SW ist es von Vorteil, wenn z.B. ein eigenes Testsystem installiert wird, das vollständig von dem operativen System entkoppelt ist und trotzdem ein kontrollierter Austausch von Daten möglich ist.

Dazu wird eine weitere externe Schnittstelle **KExDaV** spezifiziert, die neben der Datenverteiler-Datenverteiler-Kopplung eine Möglichkeit zur Verfügung stellt, bestimmte Daten zwischen zwei ansonsten völlig unabhängigen Datenverteilerbasierten Systemen austauschen zu können.

Die V-Modell Dokumentation der SW-Einheit **KExDaV** erfolgt in den weiteren Kapiteln.

3 Anwenderforderungen an die SW-Einheit "KEx-DaV"

Für die Anwenderforderungen der SW-Einheit KExDaV wird als Anwendungsfall die Integration einer neuen Applikation **App_j** in das Datenverteilersystem Neue VRZ Leverkusen betrachtet.

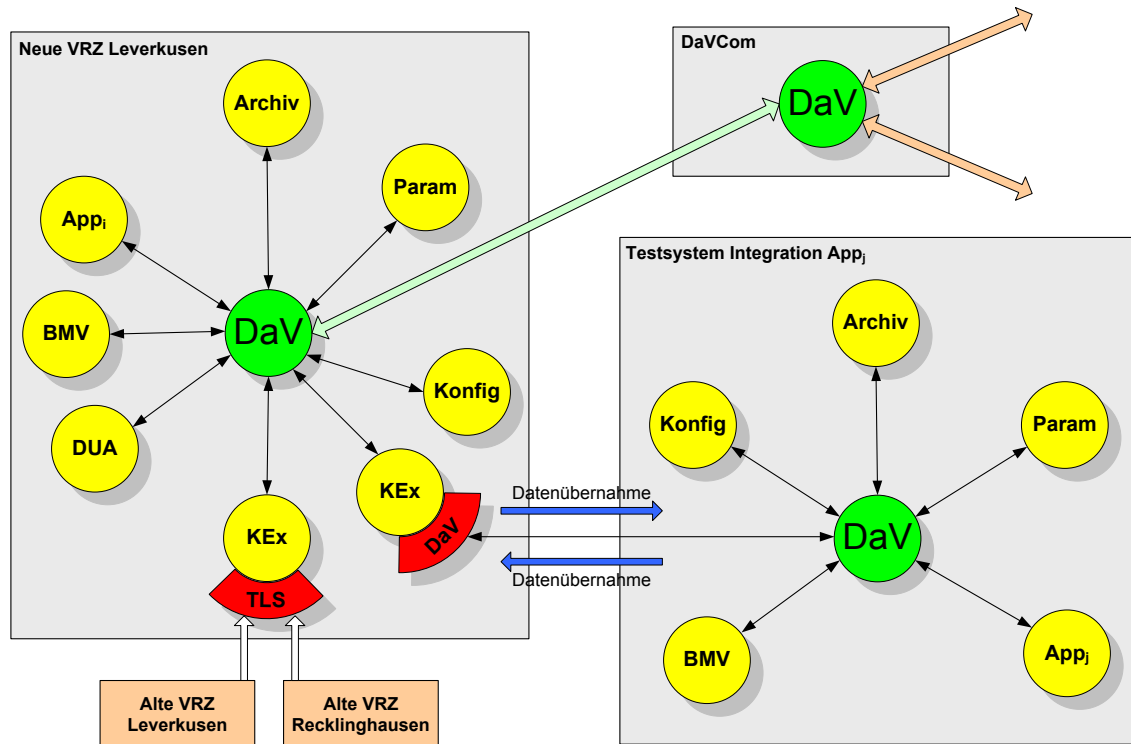


Abbildung 3-1: Integration einer neuen Applikation

Abbildung 3-1 skizziert einen Anwendungsfall der SW-Einheit KExDaV. Das operative Datenverteilersystem Neue VRZ Leverkusen läuft im "Normalbetrieb". D.h. alle SW-Einheiten und Applikation sind gestartet und die Datenverteiler-Datenverteiler-Verbindungen werden gemäß Konfiguration aufgebaut.

Für das Testsystem zur Integration der neuen SW-Einheit kann ein eigenes Datenverteilersystem gestartet werden, das alle für den Test erforderlichen Applikationen enthält. Die erforderliche Konfiguration und Parametrierung können beispielsweise initial von der Neuen VRZ Leverkusen kopiert und den Erfordernissen der Testumgebung angepasst werden. Dazu muss je Testsystem ein eigener Konfigurationsverantwortlicher eingerichtet werden.

Die zu testende Applikation **App_j** benötigt i.d.R. Daten aus dem operativen System, um die Funktionalität zu prüfen. Eventuell sollen die Ergebnisdaten der Testapplikation ab einem bestimmten Entwicklungsstand dem operativen System zur Verfügung gestellt werden.

Dabei ist zu gewährleisten, dass der operative Betrieb der Neuen VRZ Leverkusen durch das Testsystem nicht gestört werden darf.

Zur Kopplung zwischen dem Testsystem und der Neuen VRZ Leverkusen wird eine neue SW-Einheit KExDaV implementiert.

Das System, auf dem KExDaV gestartet wird und dessen Konfiguration und Parametrierung den Datenaustausch festlegt, wird als Lokal-System bezeichnet. Das andere System wird als Remote-System bezeichnet.

Afo-1

Durch die SW-Einheit KExDaV muss zwischen zwei Datenverteilerbasierten Systemen ein parametrierbarer **Datenaustausch** ermöglicht werden. Dabei müssen die beiden Systeme (z.B. die Neue VRZ Leverkusen und ein Testsystem) **völlig voneinander unabhängig** sein¹.

Folgende fachliche Anforderungen sind zu erfüllen:

- Austausch von **Online-Datensätzen**
Zwischen dem Lokal- und Remotesystem müssen in beide Richtungen Online-Datensätze zu beliebigen Attributgruppen / Aspektkombinationen ausgetauscht werden können.
- Austausch von **Parametern**
Zwischen dem Lokal- und Remotesystem müssen in beide Richtungen Parameterdaten ausgetauscht werden können.
- Austausch von **dynamischen Objekten**
Zwischen dem Lokal- und Remotesystem müssen in beide Richtungen dynamische Objekte ausgetauscht werden können.
- Austausch/Synchronisierung von **dynamischen Mengen**
Zwischen dem Lokal- und Remotesystem müssen in beide Richtungen dynamischen Mengen synchronisiert werden können. Dabei müssen ggfl. dynamische Objekte ausgetauscht werden.

Afo-2

Welche der in Afo-1 spezifizierten Daten bzw. Objekte wie ausgetauscht werden sollen, muss parametrierbar sein.

Afo-3

KExDaV muss tolerant mit unterschiedlichen Konfigurationsständen umgehen können. Die beiden zu koppelnden Systeme können unterschiedliche Versionen von Konfigurationsbereichen besitzen bzw. können Konfigurationsbereiche nur in einem System vorhanden sein. Bei dem Austausch zwischen den Systemen muss KExDaV die möglichen Korrespondenzen auflösen und entsprechend agieren. Wenn in einem System kein korrespondierendes Objekt für eine Aktion ermittelt werden kann, ist eine entsprechende Meldung auszugeben. KExDaV darf sich hierbei aber nicht beenden.

Beispiel

Wenn im Lokal-System und im Remote-System unterschiedliche Versionen einer Attributgruppe vorhanden sind und vom Remote-System Datensätze zum Lokal-System übertragen werden sollen, werden alle möglichen Daten übertragen. Wenn die Attributgruppe im Lokal-System bestimmte Attribute nicht beinhaltet, werden die entsprechenden Daten vom Remote-System ignoriert. Wenn die Attributgruppe im Remote-System bestimmte Attribute nicht beinhaltet, werden die entsprechenden Daten im Lokal-System mit den Defaultwerten besetzt.

¹ Bei einer Datenverteiler-Datenverteiler Kopplung zwischen den beiden Systemen würde sich ein Gesamtsystem ergeben, in dem sich die Systeme so verhalten, als ob nur ein logischer Datenverteiler vorhanden ist. Die Systeme verschmelzen sozusagen zu einem System mit den wichtigen Paradigmen, dass es für eine Datenidentifikation nur eine Quelle bzw. Senke geben kann. Bei der Entkopplung bzw. völligen Unabhängigkeit der Systeme können in beiden Systemen parallel "gleiche" Quellen existieren.

4 Technische Anforderungen an die SW-Einheit "KEx-DaV"

Identifikation des Elements

Die SW-Einheit "KEx-DaV" ist dem Segment 2 "Kommunikation mit externen Stellen" zugeordnet.

Die Bezeichnung der SW-Einheit im Segment 2 ist KEx-DaV.

4.1 Gesamtfunktion des Elements

4.1.1 Allgemeine Technische Anforderungen

TAnf-KExDaV 1: Allgemeine Technische Anforderungen

KExDaV muss die strikte Trennung zwischen den Konfigurationen des Lokal-Systems und des Remote-Systems sicherstellen. Insbesondere dürfen keine Objekte, die auf einer Seite (z.B. dem Lokal-System) ermittelt wurden, auf der anderen Seite (dem Remote-System) verwendet werden.

Die Ermittlung von korrespondierenden Objekten bei den jeweiligen Konfigurationen erfolgt über die PID der entsprechenden Konfigurationsobjekte oder dynamischen Objekte. Falls ein dynamisches Objekt kein PID aufweist, wird geprüft, ob bereits ein Objekt mit entsprechender ID ausgetauscht wurde. Dies kann mit Hilfe eines konfigurierenden Datensatzes überprüft werden.

Bei der Datenübertragung müssen alle benötigten Attribute umkopiert werden (D.h. es darf kein Datensatzobjekt von der einen Seite direkt an die andere Seite weitergegeben werden. Diese Anforderung ergibt sich direkt aus der Anforderung zur strikten Trennung der Konfigurationen).

Fehler bzw. nicht auflösbare Zuordnungen sind grundsätzlich durch aussagekräftige Debugausgaben und je nach Schwere zusätzlich als Betriebsmeldungen mitzuteilen.

4.1.2 Start und Initialisierung von KExDaV

Die SW-Einheit KExDaV wird auf einem System, im Folgenden Lokal-System genannt, gestartet.

Abbildung 4-1 skizziert ein mögliches Szenario, in dem zwei Datenverteilerbasierte Systeme das Lokal- und das Remote-System über die SW-Einheit KExDaV miteinander gekoppelt werden.

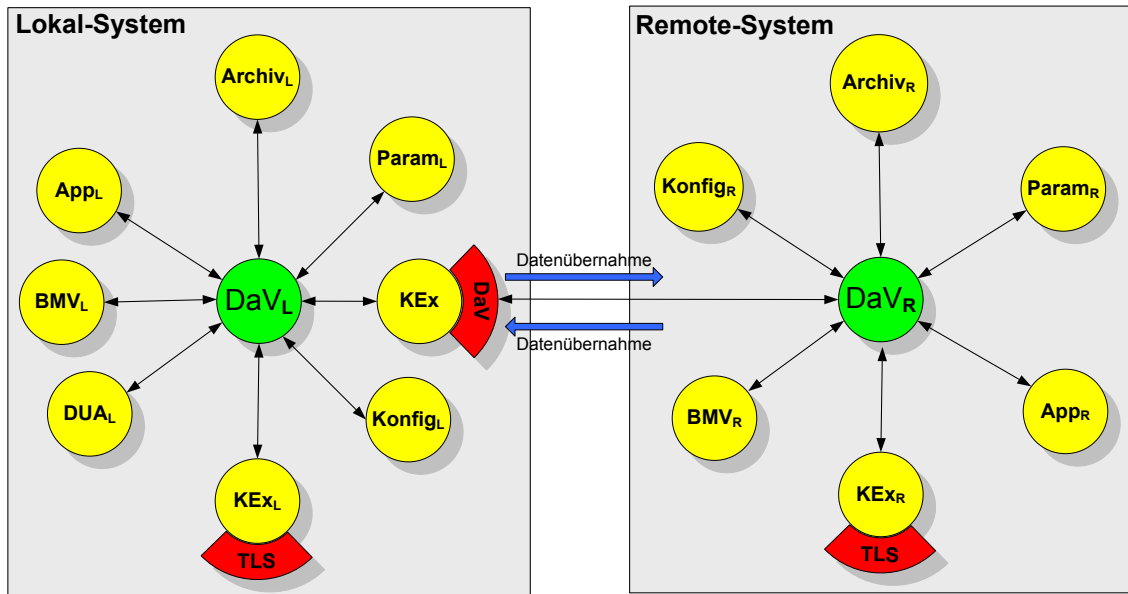


Abbildung 4-1: Kopplung zweier Datenverteilerbasierter Systeme mittels KExDaV

TAnf-KExDaV 2: Start und Initialisierung

Beim Start der SW-Einheit KExDaV werden die Standard-Aufrufargumente der Datenverteiler-Applikationsfunktionen verwendet. Im Beispiel aus Abbildung 4-1 muss z.B. für KExDaV als Aufrufargument `-datenverteiler` die IP-Adresse und Portnummer zum Datenverteiler DaV_L angegeben werden.

Pro Datenverteilerbasiertem System können mehrere SW-Einheiten KExDaV eingesetzt werden. Dazu ist als zusätzlicher Aufrufparameter `-kexDaV=PidDesKexDaV` zu ergänzen, über den der KExDaV-Applikation mitgeteilt wird, an welchem Objekt der Parameter zur Spezifikation des KExDaV "hängt"².

KExDaV muss die Abbildung von Datensätzen von einer in eine andere Attributgruppe unterstützen. Dazu ist eine Plugin-Schnittstelle vorzusehen s.a. Kapitel 4.1.5 "Plugin-Schnittstelle für den Austausch von Datensätzen zu Attributgruppen". Die vorhandenen Plugin werden KExDaV beim Start über Aufrufargumente `-plugin=KlasseDesPlugin` zur Verfügung gestellt.

² Dazu muss ein zusätzlicher Typ KExDaV eingerichtet werden, an dem der Parameter zur Spezifikation der KExDaV-Applikation hängt. Damit im Normalfall kein eigenes Objekt zum Typ KExDaV versorgt werden muss, erweitert der Typ AutarkeOrganisationsEinheit den neuen Typ KExDaV.

Nachdem KExDaV die Verbindung zum Lokal-System aufgebaut hat, muss sie sich auf den Parameterdatensatz *SpezifikationKExDaV* anmelden, in dem spezifiziert ist, welche Remote-DaV Systeme zu kontaktieren sind und welche Daten mit den einzelnen Remote-DaV-Systemen ausgetauscht werden sollen. Dieser Parameterdatensatz ist dem lokalen Datenverteiler DaV_L zugeordnet. Abbildung 4-2 skizziert den Aufbau des Parameterdatensatzes *SpezifikationKExDaV*. Der Parameter besteht auf oberster Ebene aus einer Attributliste *RemoteDaV*, in dem der Datenverteiler eines Remote-Systems angegeben ist. Die Verbindungsparameter zu dem Remote-Datenverteiler werden durch die Attribute

IP-Adresse	IP-Adresse, unter der der Remote-Datenverteiler erreichbar ist.
Portnummer	Portnummer, unter der der Remote-Datenverteiler Applikationsverbindungen annimmt.
Benutzer	Benutzer, unter dem sich KExDaV bei dem Remote-Datenverteiler anmeldet.
Wartezeit	Über diesen Parameter wird vorgegeben, wie lange der lokale Datenverteiler nach einem Verbindungsabbruch zu dem entsprechenden Remote-Datenverteiler warten soll, bevor ein erneuter Verbindungsaufbau erfolgt.
DaV-Pid	PID des Remote-Datenvertailers.
Aktiv	Über diesen Schalter lässt sich einstellen, ob der Austausch mit dem angegebenen Remote-Datenverteiler aktiv ist (Ja/Nein). Hierüber lässt sich eine Verbindung gezielt abschalten, ohne dass die Vorgaben im Parameter verloren gehen.

Parameterattributgruppe Spezifikation KExDaV									
RemoteDaV[...]									
IP-Adresse		Portnummer		RemoteAustauschDefault			LokalAustauschDefault		
Benutzer				RemoteAustauschBereiche[...]			LokalAustauschBereiche[...]		
Wartezeit		DaV-Pid	Aktiv	Konfigurationsbereich		Typ[...]	Konfigurationsbereich		Typ[...]
AustauschOnlinedaten[...]									
AuswahlBereich[...]				AuswahlRegion[...]			AuswahlObjekte[...]		
KV[...]	KB[...]	Typ[...]	Menge	Region[...]	Typ[...]	Menge	Objekte[...]	Menge	
Onlinedaten[...]									
AtgLokal	AtgRemote	AspektLokal	AspektRemote	SimLokal	SimRemote	Delta	Nachgeliefert	Richtung	
AustauschParameterdaten[...]									
AuswahlBereich[...]				AuswahlRegion[...]			AuswahlObjekte[...]		
KV[...]	KB[...]	Typ[...]	Menge	Region[...]	Typ[...]	Menge	Objekte[...]	Menge	
Parameterdaten[...]									
Attributgruppen[...]		SimLokal		SimRemote		Delta		Strategie	
AtgLokal	AtgRemote								
AustauschDynamischeObjekteLokalNachRemote[...]									
AuswahlBereichTypen[...]					AuswahlRegionTypen[...]				
KV[...]		KB[...]		Typ[...]		Region[...]		Typ[...]	
AustauschDynamischeObjekteRemoteNachLokal[...]									
AuswahlBereichTypen[...]					AuswahlRegionTypen[...]				
KV[...]		KB[...]		Typ[...]		Region[...]		Typ[...]	
AustauschDynamischeMengen[...]									
DynamischeMengeLokal			DynamischeMengeRemote			Richtung			
Objekt		Menge	Objekt		Menge				

Abbildung 4-2: Parameterattributgruppe SpezifikationKExDaV

Weitere allgemeine Parameter für die Verbindung zwischen dem Lokal-System und dem Remote-System sind die Konfigurationsbereiche, in denen ausgetauschte dynamische Objekte gespeichert werden:

`RemoteAustauschBereichDefault`

Konfigurationsbereich, der von der Konfiguration des Remote-Systems geladen werden muss. In diesem Bereich werden i.d.R. dynamische Objekte, die vom Lokal-System auf das Remote-System übertragen werden sollen, gespeichert. Dabei muss bei der Einrichtung sichergestellt werden, dass KExDaV dynamische Objekte in diesem Bereich im Remote-System anlegen, ändern und löschen kann.

`LokalAustauschBereichDefault`

Konfigurationsbereich, der von der Konfiguration des Lokal-Systems geladen werden muss. In diesem Bereich werden i.d.R. dynamische Objekte, die vom Remote-System auf das Lokal-System übertragen werden sollen, gespeichert. Dabei muss bei der Einrichtung sichergestellt werden, dass KExDaV dynamische Objekte in diesem Bereich im Lokal-System anlegen, ändern und löschen kann.

`RemoteAustauschBereiche[]`

Attributliste, über die weitere Konfigurationsbereiche angegeben werden können, in denen dynamische Objekte vorgegebener Typen, die vom Lokal-System auf das Remote-System übertragen werden sollen, gespeichert werden. Diese Liste kann leer sein. In diesem Fall wird nur der Defaultbereich benutzt. Auch die hier aufgeführten Konfigurationsbereiche müssen von der Konfiguration des Remote-Systems geladen werden. Dabei muss bei der Einrichtung sichergestellt werden, dass KExDaV dynamische Objekte in diesen Bereichen im Remote-System anlegen, ändern und löschen kann.

`LokalAustauschBereiche[]`

Attributliste, über die weitere Konfigurationsbereiche angegeben werden können, in denen dynamische Objekte vorgegebener Typen, die vom Remote-System auf das Lokal-System übertragen werden sollen, gespeichert werden. Diese Liste kann leer sein. In diesem Fall wird nur der Defaultbereich benutzt. Auch die hier aufgeführten Konfigurationsbereiche müssen von der Konfiguration des Lokal-Systems geladen werden. Dabei muss bei der Einrichtung sichergestellt werden, dass KExDaV dynamische Objekte in diesen Bereichen im Lokal-System anlegen, ändern und löschen kann.

KExDaV muss den Parameter `SpezifikationKExDaV` unter dem Aspekt `Ist` publizieren.

4.1.2.1 Verbindungsaufbau zu den Remote-Datenverteilern

TAnf-KExDaV 3: Verbindungsaufbau Remote DaV

Pro Datenverteilerbasiertem System können mehrere SW-Einheiten KExDaV eingesetzt werden. Die Parametrierung der einzelnen KExDaV wird über den Aufrufparameter `-kexDaV` gesteuert. Dabei soll jedes KExDaV beliebig viele Remote-Systeme bearbeiten können. Deshalb ist die Attributliste `RemoteDaV` als Feld modelliert.

DaV-KEx muss zu jedem spezifizierten Remote-Datenverteiler eine eigene Datenverteiler-Verbindung aufbauen.

- Das für die Authentifizierung beim Remote-Datenverteiler zu verwendende Passwort wird aus der üblichen Passwortdatei gelesen, die über das Standard-Argument

-authentifizierung angegeben wurde. In der Passwortdatei wird ein Eintrag der Form Benutzer@Dav-Pid: `passwort` gesucht. Der entsprechende Benutzer und die Datenverteiler-Pid sind in dem Parameter aufgeführt.

- Nach dem Verbindungsaufbau zu einem Remote-System muss von KEx-DaV sichergestellt werden, dass die im Parameterdatensatz angegebene Datenverteiler-Pid mit der tatsächlichen Pid des Remote-Datenvertellers übereinstimmt.
- Wenn die Verbindung zu einem Remote-Datenverteiler nicht zustandekommt oder verloren geht, dann muss KEx-DaV nach einer parametrierbaren Wartezeit erneut versuchen, die Verbindung wieder herzustellen.

4.1.2.2 Abbruch der Datenverteilerverbindung zum Lokal-System

TAnf-KExDaV 4: Verbindungsabbruch Lokal-System

Wenn die Verbindung zum lokalen DaV-System verloren geht, dann muss KEx-DaV sich beenden.

4.1.3 Austausch von Onlinedaten

TAnf-KExDaV 5: Austausch von Onlinedaten

Um Onlinedaten zwischen dem Lokalen und den Remote System auszutauschen, muss die SW-Einheit KExDaV bei dem Quellsystem die Daten besorgen und in das Zielsystem einspeisen. Da die Daten in beide Richtungen ausgetauscht werden sollen, kann das Lokal-System sowohl Quell- als auch Zielsystem sein.

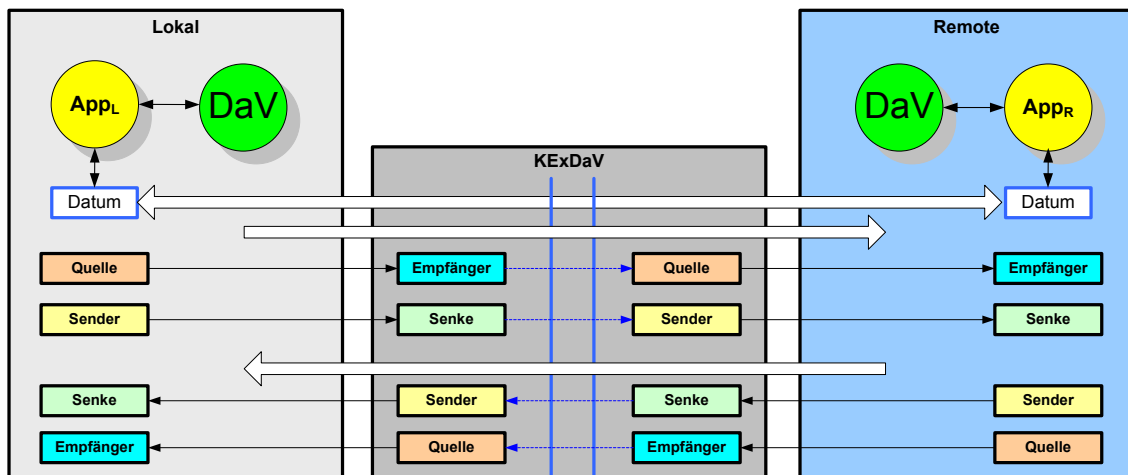


Abbildung 4-3: Austausch von Onlinedaten

Abbildung 4-3 zeigt die verschiedenen Möglichkeiten, die für den Datenaustausch zur Verfügung stehen müssen.

- Datenaustausch vom Lokal-System zum Remote-System
 - **Lokal Quelle → Remote Empfänger**
 Für diesen Fall muss sich KExDaV beim Datenverteiler des Lokal-Systems als Empfänger anmelden und beim Datenverteiler des Remote-Systems als Quelle.

- **Lokal Sender → Remote Senke**
Für diesen Fall muss sich KExDaV beim Datenverteiler des Lokal-Systems als Senke anmelden und beim Datenverteiler des Remote-Systems als Sender.
- Datenaustausch vom Remote-System zum Lokal-System
- **Lokal Senke ← Remote Sender**
Für diesen Fall muss sich KExDaV beim Datenverteiler des Lokal-Systems als Sender anmelden und beim Datenverteiler des Remote-Systems als Senke.
- **Lokal Empfänger ← Remote Quelle**
Für diesen Fall muss sich KExDaV beim Datenverteiler des Lokal-Systems als Quelle anmelden und beim Datenverteiler des Remote-Systems als Empfänger.

Die Spezifikation, welche Onlinedatensätze für welche Konfigurationsobjekte in welcher Richtung wie ausgetauscht werden, erfolgt über den Parameter `SpezifikationKExDaV` (s.a. Abbildung 4-2). Dafür steht die Attributliste `AustauschOnlinedaten` zur Verfügung, in der zunächst die zu behandelnden Konfigurationsobjekte und danach die gewünschten Onlinedaten vorgegeben werden. Damit hier mehrere Vorgaben getätigt werden können, ist die Attributliste `AustauschOnlinedaten` als Feld modelliert.

4.1.3.1 Konfigurationsobjektauswahl

TAnf-KExDaV 6: Konfigurationsobjektauswahl

Bei der Spezifikation muss festgelegt werden, für welche Konfigurationsobjekte die Daten ausgetauscht werden sollen. Hierzu sind gemäß Parameter (s.a. Abbildung 4-4) folgende Möglichkeiten zu unterstützen:

AuswahlBereich[..]				AuswahlRegion[..]			AuswahlObjekte[..]	
KV[..]	KB[..]	Typ[..]	Menge	Region[..]	Typ[..]	Menge	Objekte[..]	Menge

Abbildung 4-4: Konfigurationsobjektauswahl

- **AuswahlBereich[0..)**
Attributliste als Array der Größe 0 bis unbegrenzt. Wenn dieses Array leer ist, erfolgt (hierüber) keine Auswahl von Konfigurationsobjekten.
- **Array Konfigurationsverantwortliche [0 ..).**
Über dieses Array können Konfigurationsverantwortliche angegeben werden. Die Auswahl eines Konfigurationsverantwortlichen bedeutet, dass alle Konfigurationsbereiche dieses Konfigurationsverantwortlichen betrachtet werden.
Wenn das Array Konfigurationsverantwortliche leer ist, ist kein Konfigurationsverantwortlicher und damit kein Konfigurationsbereich ausgewählt.
- **Array Konfigurationsbereich [0 ..).**
Über dieses Array können Konfigurationsbereiche angegeben werden. Die Auswahl eines Konfigurationsbereichs bedeutet, dass alle Konfigurationsobjekte dieses Konfigurationsbereichs betrachtet werden.
Wenn das Array Konfigurationsbereich leer ist, ist kein Konfigurationsbereich ausgewählt.
- **Array Typen [0 ..).**
Über dieses Array werden die bisher ausgewählten Objekte auf die angegebenen Typobjekte beschränkt. Wenn die beiden Arrays Konfigurationsverantwortliche und Konfigurationsbereich beide zu keiner (Vor)Auswahl von Konfigurationsbereichen geführt haben, wird an dieser Stelle die gesamte Konfiguration betrachtet.

D.h., wenn hier z.B. als Typ MessQuerschnitt angegeben wird, werden nur noch Konfigurationsobjekte betrachtet, die von diesem Typ stammen.

Wenn das Array Typen leer ist, erfolgt keine Filterung nach Objekttypen. Damit sind alle bisher ausgewählten Konfigurationsobjekte ausgewählt (Also entweder alle Konfigurationsobjekte der (vor)ausgewählten Bereiche oder alle Konfigurationsobjekte der Konfiguration).

- **Menge**
Über die Vorgabe eines Mengennamens können die Objekte ausgewählt werden, die in Mengen dieses Namens bei den ausgewählten Objekten enthalten sind. Ist hier z.B. als Menge "FahrStreifen" angegeben, wird für alle bisher ausgewählten Objekte geprüft, ob an dem Konfigurationsobjekt eine Menge dieses Namens vorhanden ist und für diesen Fall werden die enthaltenen Konfigurationsobjekte betrachtet.
Wenn hier keine Angabe erfolgt, bleibt die Auswahl bestehen.
- **AuswahlRegion[0..]**
Attributliste als Array der Größe 0 bis unbegrenzt. Wenn dieses Array leer ist, erfolgt (hierüber) keine Auswahl von Konfigurationsobjekten.
 - **Array Region [0 ..].**
Über dieses Array können bereits definierte Regionen in der Konfiguration angegeben werden. Die Auswahl einer Region bedeutet, dass alle Konfigurationsobjekte dieser Region betrachtet werden.
Wenn das Array Region leer ist, sind alle Konfigurationsobjekte ausgewählt.
 - **Array Typen [0 ..]**
Über dieses Array werden die bisher ausgewählten Objekte auf die angegebenen Typobjekte beschränkt. D.h., wenn hier z.B. als Typ MessQuerschnitt angegeben wird, werden nur noch Konfigurationsobjekte betrachtet, die von diesem Typ stammen.
Wenn hier keine Angabe erfolgt, bleibt die Auswahl bestehen.
 - **Menge**
Über die Vorgabe eines Mengennamens können die Objekte ausgewählt werden, die in Mengen dieses Namens bei den ausgewählten Objekten enthalten sind. Ist hier z.B. als Menge "FahrStreifen" angegeben, wird für alle bisher ausgewählten Objekte geprüft, ob an dem Konfigurationsobjekt eine Menge dieses Namens vorhanden ist und für diesen Fall werden die enthaltenen Konfigurationsobjekte betrachtet
- **AuswahlObjekte[0..]**
Attributliste als Array der Größe 0 bis unbegrenzt. Wenn dieses Array leer ist, erfolgt (hierüber) keine Auswahl von Konfigurationsobjekten.
 - **Array Objekte [0 ..]**
Über dieses Array können beliebige Konfigurationsobjekte angegeben werden.
Bei einem leeren Array sind alle Konfigurationsobjekte ausgewählt.
 - **Menge**
Über die Vorgabe eines Mengennamens können die Objekte ausgewählt werden, die in Mengen dieses Namens bei den ausgewählten Objekten enthalten sind. Ist hier z.B. als Menge "FahrStreifen" angegeben, wird für alle bisher ausgewählten Objekte geprüft, ob an dem Konfigurationsobjekt eine Menge dieses Namens vorhanden ist und für diesen Fall werden die enthaltenen Konfigurationsobjekte betrachtet.

4.1.3.1.1 Ermittlung korrespondierender Konfigurationsobjekte

TAnf-KExDaV 7: Korrespondierende Konfigurationsobjekte

Für jedes durch Kapitel 4.1.3.1 ermittelte Konfigurationsobjekt wird im Lokal-System und im Remote-System das korrespondierende Objekt in der Konfiguration bestimmt.

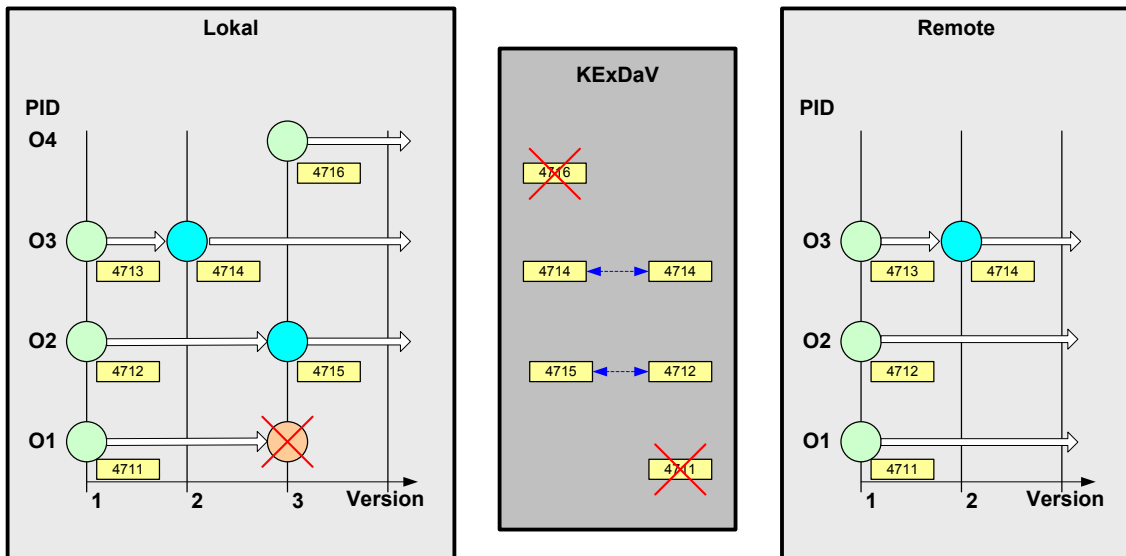


Abbildung 4-5: Unterschiedliche Konfigurationsstände

Abbildung 4-5 skizziert die Möglichkeiten, die bei unterschiedlichen Konfigurationsständen vorliegen können. In dem Beispiel arbeiten das Lokal-System mit der Version 3 und das Remote-System mit der Version 2 eines betrachteten Konfigurationsbereichs. Die Konfigurationsobjekte mit den **PIDs** o1, o2, o3 und o4 sollen laut Konfigurationsobjektauswahl alle beim Datenaustausch berücksichtigt werden. Objektentstehungen und versionspflichtige Objektänderungen sind in Abbildung 4-5 durch Kreis gekennzeichnet. So werden die Konfigurationsobjekte o1, o2 und o3 in Version 1 angelegt und erhalten die in den gelben Rechtecken aufgeführten IDs. Konfigurationsobjekt o2 wird in Version 2 geändert und erhält die neue ID 4714. Konfigurationsobjekt o3 ist ab Version 3 nicht mehr gültig. Außerdem wird in Version 3 Konfigurationsobjekt o2 versionspflichtig geändert und o4 entsteht in dieser Version.

KExDaV ermittelt für das Lokal-System die gültigen Objekte mit den IDs 4715 (PID o2), 4714 (o3) und 4716 (o4). Für die PID o1 kann kein gültiges Konfigurationsobjekt ermittelt werden, da dieses Objekt ab Version 3 nicht mehr gültig ist. Für das Remote-System werden analog die gültigen Objekte mit den IDs 4711 (o1), 4712 (o2) und 4714 (o3) ermittelt. Für die PID o4 kann kein gültiges Konfigurationsobjekt ermittelt werden, da dieses Objekt erst ab Version 3 gültig ist. Als Ergebnis wurden in dem Beispiel die beiden korrespondierenden Objekte o2 zu und o3 ermittelt.

Für alle korrespondierenden gültigen Konfigurationsobjekte muss KExDaV die Datenübertragung gewährleisten. Damit werden z.B. für das Konfigurationsobjekt o2 mit der ID 4715 in Lokal-System entstehende Datensätze im Remote-System für das Konfigurationsobjekt o2 mit der ID 4712 eingespeist und umgekehrt. Für die Objekte o1 und o4 ist kein Datenaustausch möglich, da hier keine korrespondierenden Konfigurationsobjekte ermittelbar sind. Dieser Sachverhalt ist durch aussagekräftige Debugausgaben und Betriebsmeldungen zu melden.

Allgemein muss KExDaV für alle zu betrachtenden Konfigurationsobjekte und dynamischen Objekte korrespondierende gültige Objekte ermitteln.

4.1.3.2 Auswahl Onlinedaten

TAnf-KExDaV 8: Auswahl Onlinedaten

Für die Auswahl der Onlinedaten sind folgende Angaben erforderlich bzw. optional möglich:

Onlinedaten[..]								
AtgLokal	AtgRemote	AspektLokal	AspektRemote	SimLokal	SimRemote	Delta	Nachgeliefert	Richtung

Abbildung 4-6: Auswahl Onlinedaten

- **Onlinedaten[0..]**
Attributliste als Array der Größe 0 bis unbegrenzt.
 - **Attributgruppe**
Für die auszutauschenden Onlinedaten wird die Attributgruppe spezifiziert, zu der Datensätze ausgetauscht werden sollen.
Im Regelfall wird für den Austausch der Datensätze im Quell- und Remotesystem die gleiche Attributgruppe verwendet.
KExDaV muss zusätzlich die Möglichkeit bieten, dass Datensätze aus dem Quellsystem auf Datensätze zu einer anderen Attributgruppe im Zielsystem abgebildet werden. Dazu muss eine Plugin-Schnittstelle implementiert werden, die vorgibt, wie Daten von einer auf eine andere Attributgruppe umzusetzen sind. Die möglichen Plugin werden über ein zusätzliche Aufrufargumente vorgegeben.
 - **AtgLokal**
Hier wird eine Referenz auf die Attributgruppe im Lokalsystem, zu der Datensätze ausgetauscht werden sollen, angegeben.
 - **AtgRemote**
Hier kann die PID der Attributgruppe im Remotesystem, zu der Datensätze ausgetauscht werden sollen, (als Zeichenkette) angegeben werden. Als Defaultwert wird hier ein Leerstring vorgegeben, was bewirkt, dass die Daten am Ziel unter der gleichen Attributgruppe eingespeist werden.
 - **AspektLokal**
Hier wird eine Referenz auf den Aspekt zu der Attributgruppe angegeben, der in dem Lokalsystem für den Datenaustausch angemeldet werden soll.
 - **AspektRemote**
Hier kann die PID eines Aspektes zu der Attributgruppe (als Zeichenkette) angegeben werden, der in dem Remotesystem für den Datenaustausch angemeldet werden soll. Bei einer gewünschten Aspektumleitung kann an dieser Stelle ein anderer Aspekt als unter AspektLokal angegeben werden. Als Defaultwert wird hier ein Leerstring vorgegeben, was bewirkt, dass die Daten im Remotesystem unter dem gleichen Aspekt eingespeist werden.
 - **SimLokal**
Hier kann eine Simulationsvariante bei der Anmeldung im Lokalsystem angegeben werden. Als Default wird der Normalbetrieb vorbesetzt (Variante 0).

- **SimRemote**
Hier kann eine Simulationsvariante bei der Anmeldung im Remotesystem angegeben werden. Als Default wird der Normalbetrieb vorbesetzt (Variante 0).
- **Delta**
Über das Attribut Delta wird spezifiziert, ob bei dem Austausch nur geänderte Datensätze übertragen werden sollen, oder ob alle Datensätze übertragen werden sollen.
- **Nachgeliefert**
Über das Attribut Nachgeliefert wird spezifiziert, ob bei dem Austausch auch die als nachgeliefert gekennzeichneten Datensätze übertragen werden sollen oder nicht.
- **Richtung**
Über die Auswahl Richtung wird die Richtung des Datenaustausches festgelegt. Folgende Angaben sind dabei möglich (s.a. Abbildung 4-3):
 - **Lokal Quelle → Remote Empfänger**
 - **Lokal Sender → Remote Senke**
 - **Lokal Senke ← Remote Sender**
 - **Lokal Empfänger ← Remote Quelle**

4.1.3.2.1 Behandlung bei unterschiedlichen Konfigurationsständen für Attributgruppen oder Aspekte

TAnf-KExDaV 9: Unterschiedliche Konfigurationsstände

Analog zu Kapitel 4.1.3.1.1 "Ermittlung korrespondierender Konfigurationsobjekte" kann es möglich sein, dass Attributgruppen oder Aspekte in unterschiedlichen Versionsständen in den beiden Systemen vorliegen.

In diesem Fall muss KExDaV jeweils die gültige Version der Attributgruppe oder des Aspektes verwenden. Wenn eine Attributgruppe oder ein Aspekt in einem der beiden Systeme nicht gültig ist, kann der Datenaustausch zu dieser Anmeldung nicht durchgeführt werden.

Falls eine Attributgruppe in unterschiedlichen Versionen vorliegt, muss KExDaV die Daten zu den Attributen, die in beiden Attributgruppen vorhanden sind übertragen und die anderen entweder unterdrücken, wenn das entsprechende Attribut in der Zielattributgruppe nicht vorhanden ist oder mit dem Defaultwert belegen, falls das entsprechende Attribut in der Quellattributgruppe nicht vorhanden ist.

4.1.3.2.2 Behandlung von Referenzen auf Objekte in den Onlinedatensätzen

TAnf-KExDaV 10: Referenzen in Onlinedatensätzen

In Onlinedatensätzen können Referenzen auf Objekte enthalten sein. Für diese Fälle muss KExDaV die korrespondierenden Objekte ermitteln (s. Kapitel 4.1.3.1.1 "Ermittlung korrespondierender Konfigurationsobjekte").

Referenz auf ein Konfigurationsobjekt

Wenn im Zielsystem ein korrespondierendes Objekt vorhanden ist, ist eine Referenz auf dieses Konfigurationsobjekt in dem Datensatz zu übertragen.

Wenn das nicht der Fall ist, muss geprüft werden, ob das Attribut den Wert undefiniert zulässt. Falls dies der Fall ist, ist an der entsprechenden Stelle undefiniert zu übertragen.

Falls der Wert undefiniert nicht zugelassen ist, kann der Datensatz nicht übertragen werden. In diesem Fall ist ein Datensatz mit der Kennung "Keine Daten" zu übertragen und eine entsprechende Meldung auszugeben (Debug, Betriebsmeldung).

Referenz auf ein dynamisches Objekt

Wenn im Zielsystem ein korrespondierendes Objekt vorhanden ist, ist eine Referenz auf dieses Konfigurationsobjekt in dem Datensatz zu übertragen.

Wenn das nicht der Fall ist, muss KExDaV dafür sorgen, dass das entsprechende dynamische Objekt in dem Zielsystem entsprechend angelegt wird. Dies erfolgt analog zu Kapitel 4.1.6 "Austausch von dynamischen Objekten".

4.1.4 Austausch von Parameterdaten

TAnf-KExDaV 11: Austausch von Parameterdaten

Für den Austausch von Parametern muss es eine spezifische Unterstützung geben, die die Parametrierung des Austauschs vereinfacht. Es muss parametrierbar sein, für welche Objekte und für welche Attributgruppen (d.h. Datenidentifikation ohne Aspekt) Parameter ausgetauscht werden sollen. Diese Anforderung wird durch die Attributliste `AustauschParameterdaten` modelliert.

AustauschParameterdaten[..]									
AuswahlBereich[..]				AuswahlRegion[..]			AuswahlObjekte[..]		
KV[..]	KB[..]	Typ[..]	Menge	Region[..]	Typ[..]	Menge	Objekte[..]	Menge	
Parameterdaten[..]									
Attributgruppen[..]		SimLokal	SimRemote	Delta	Strategie				
AtgLokal	AtGRemote								

Abbildung 4-7: Austausch von Parameterdaten

4.1.4.1 Konfigurationsobjektauswahl

TAnf-KExDaV 12: Konfigurationsobjektauswahl

Die Konfigurationsobjektauswahl erfolgt analog zu Kapitel 4.1.3.1 "Konfigurationsobjektauswahl" unter Beachtung von Kapitel 4.1.3.1.1 "Ermittlung korrespondierender Konfigurationsobjekte".

4.1.4.2 Parameterdaten

TAnf-KExDaV 13: Parameterdaten

- Parameterdaten
Attributliste als Array der Größe 0 bis unbegrenzt.
- Attributgruppe
Hier werden die Parameterattributgruppen zu der Datensätze ausgetauscht werden sollen,

angegeben. Im Regelfall wird für den Austausch der Datensätze im Lokal- und Remote-system die gleiche Parameterattributgruppe verwendet.

KExDaV muss zusätzlich die Möglichkeit bieten, dass Datensätze aus dem Quellsystem auf Datensätze zu einer anderen Parameterattributgruppe im Zielsystem abgebildet werden. Dazu muss eine Plugin-Schnittstelle implementiert werden, die vorgibt, wie Daten von einer auf eine andere Attributgruppe umzusetzen sind. Die möglichen Plugin werden über ein zusätzliche Aufrufargumente vorgegeben.

- **AtgLokal**
Hier wird eine Referenz auf die Parameterattributgruppe im Lokalsystem, zu der Datensätze ausgetauscht werden sollen, angegeben.
- **AtgRemote**
Hier kann die PID der Parameterattributgruppe im Remotesystem, zu der Datensätze ausgetauscht werden sollen, (als Zeichenkette) angegeben werden. Als Defaultwert wird hier ein Leerstring vorgegeben, was bewirkt, dass die Daten am Ziel unter der gleichen Parameterattributgruppe eingespeist werden.
- **SimLokal**
Hier kann eine Simulationsvariante bei der Anmeldung im Lokalsystem angegeben werden. Als Default wird der Normalbetrieb vorbesetzt (Variante 0).
- **SimRemote**
Hier kann eine Simulationsvariante bei der Anmeldung im Remotesystem angegeben werden. Als Default wird der Normalbetrieb vorbesetzt (Variante 0).
- **Delta**
Über das Attribut Delta wird spezifiziert, ob bei dem Austausch nur geänderte Datensätze übertragen werden sollen, oder ob alle Datensätze übertragen werden sollen.
- **Strategie**
Hier kann eine Strategie für den Parameternaustausch gewählt werden. Dabei sind folgende Strategien möglich:
 - Verwaltung der Parameter im Lokal-System (Remote-System nur Lesen)
 - Verwaltung der Parameter im Lokal-System (Remote-System Lesen und Ändern)
 - Verwaltung der Parameter im Remote-System (Lokal-System nur Lesen)
 - Verwaltung der Parameter im Remote-System (Lokal-System Lesen und Ändern)
 - Verwaltung der Parameter im Lokal- und Remote-System
 - Verwaltung der Parameter im Lokal- und Remote-System Austausch nur bei Triggersignal

In den weiteren Unterpunkten werden die unterschiedlichen Strategien detailliert erläutert.

4.1.4.2.1 Verwaltung der Parameter im Lokal-System (Remote-System nur Lesen)

TAnf-KExDaV 14: Verwaltung Parameter Lokal-System (r)

Bei dieser Strategie erfolgt die Verwaltung der Parameter ausschließlich im Lokal-System. Die Parameter können im Remote-System nur gelesen und nicht verändert werden.

In Abbildung 4-8 wird diese Strategie skizziert. Die ausgewählten Parameterdaten werden nur in dem Lokal-System verwaltet³.

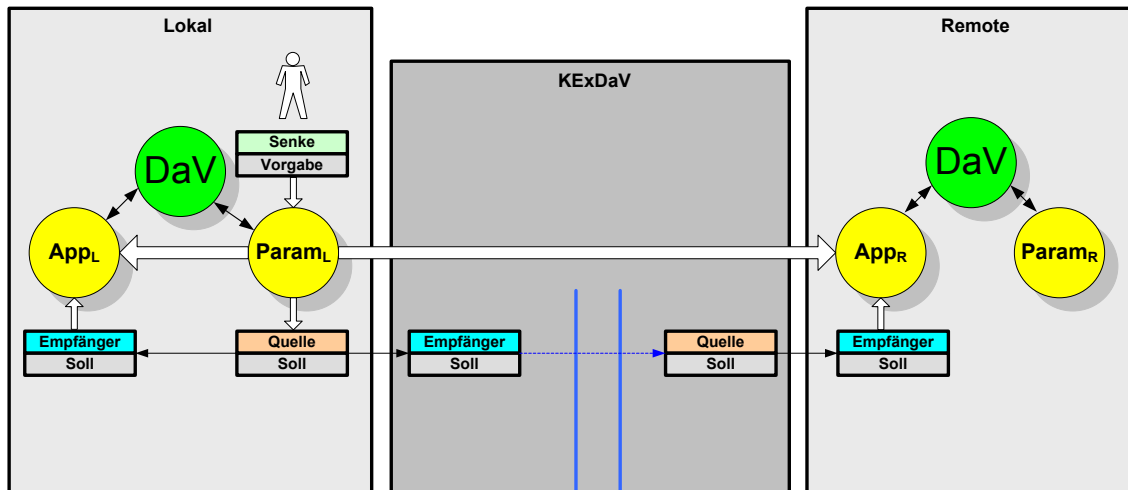


Abbildung 4-8: Strategie Verwaltung Lokal-System (nur Lesen)

Damit Applikationen im Remote-System ebenfalls auf diese Parameterdaten zugreifen können, muss KExDaV dafür sorgen, dass die entsprechenden Sollparameter (entsprechende Parameterattributgruppen unter dem Aspekt Soll) in das Remote-System übertragen werden. Dazu muss KExDaV sich für die entsprechenden Parameterattributgruppen als Empfänger für den Aspekt Soll beim Lokal-System anmelden und diese Daten als Quelle den Remote-System zur Verfügung stellen. Dabei sind die Angaben der Simulationsvarianten zu berücksichtigen. Außerdem muss die Vorgabe zum Attribut Delta entsprechend berücksichtigt werden. Wenn an dieser Stelle der Wert "Ja" gesetzt ist, werden die Parameterdaten nur bei Änderung übertragen.

³ Damit die ausgewählten Parameterdaten im Lokal-System verwaltet werden, muss natürlich die Parametrierung der Parametrierungsapplikation $Param_L$ so eingestellt sein, dass sie für die entsprechenden Parameter zuständig ist. Weiter muss Parametrierung der Parametrierungsapplikation $Param_R$ so eingestellt sein, dass sie für die entsprechenden Parameter nicht zuständig ist.

4.1.4.2.2 Verwaltung der Parameter im Lokal-System (Remote-System Lesen und Ändern)

TAnf-KExDaV 15: Verwaltung Parameter Lokal-System (rw)

Bei der in Kapitel 4.1.4.2.1 beschriebenen Strategie können die Parameter nur im Lokal-System durch z.B. den GTM geändert werden, da keine Möglichkeit zur Übertragung der Vorgabeparameter aus dem Remote-System besteht.

Die in Abbildung 4-9 skizzierte Strategie ermöglicht, Vorgaben aus dem Remote-System im Lokal-System einzubringen.

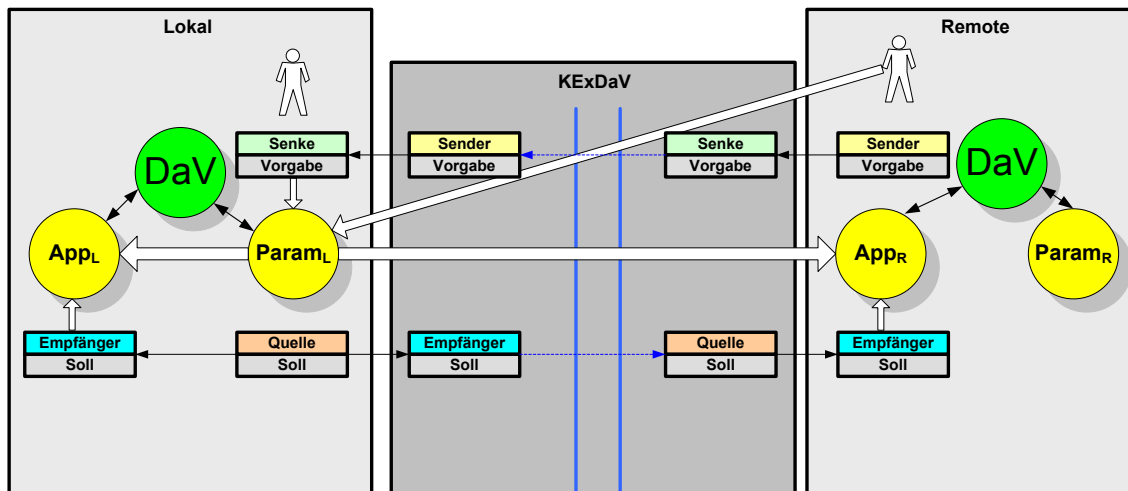


Abbildung 4-9: Strategie Verwaltung Lokal-System (Lesen und Ändern)

Als Erweiterung zur vorherigen Strategie muss KExDaV dafür sorgen, dass die entsprechenden Vorgabeparameter (entsprechende Parameterattributgruppen unter dem Aspekt Vorgabe) in das Lokal-System übertragen werden. Dazu muss KExDaV sich für die entsprechenden Parameterattributgruppen als Senke für den Aspekt Vorgabe beim Remote-System anmelden und diese Daten als Sender dem Lokal-System zur Verfügung stellen. Dabei sind die Angaben der Simulationsvarianten zu berücksichtigen. Außerdem muss die Vorgabe zum Attribut Delta entsprechend berücksichtigt werden. Wenn an dieser Stelle der Wert "Ja" gesetzt ist, werden die Parametervorgabedaten nur bei Änderung übertragen.

4.1.4.2.3 Verwaltung der Parameter im Remote-System (Lokal-System nur Lesen)

TAnf-KExDaV 16: Verwaltung Parameter Remote-System (r)

Bei dieser Strategie werden die Parameter von der Parametrierung im Remote-System verwaltet. Die Parameter können auch im Lokal-System gelesen aber nicht geändert werden. Die Vorgehensweise ist damit analog zu der in Kapitel 4.1.4.2.1 "Verwaltung der Parameter im Lokal-System (Remote-System nur Lesen)" beschriebenen Strategie umzusetzen.

4.1.4.2.4 Verwaltung der Parameter im Remote-System (Lokal-System Lesen und Ändern)

TAnf-KExDaV 17: Verwaltung Parameter Remote-System (rw)

Bei dieser Strategie werden die Parameter von der Parametrierung im Remote-System verwaltet. Die Parameter können auch im Lokal-System gelesen und geändert werden. Die Vorgehensweise ist damit analog zu der in Kapitel 4.1.4.2.2 "Verwaltung der Parameter im Lokal-System (Remote-System Lesen und Ändern)" beschriebenen Strategie umzusetzen.

4.1.4.2.5 Verwaltung der Parameter im Lokal- und Remote-System

TAnf-KExDaV 18: Verwaltung Parameter Lokal- und Remote-System

Bei dieser Strategie erfolgt die Verwaltung der Parameter in beiden Systemen. Änderungen der Parameter werden synchronisiert, d.h. Parameteränderungen in einem System werden vom jeweils anderen System übernommen⁴.

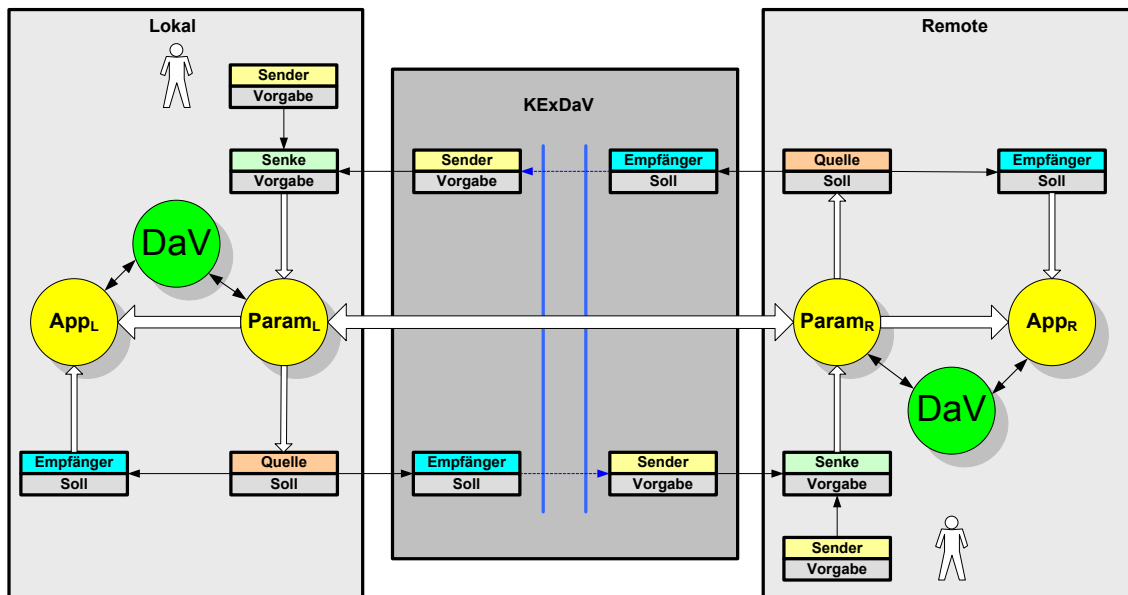


Abbildung 4-10: Strategie der Parameter im Lokal- und Remote-System

Abbildung 4-10 skizziert die Abläufe bei dieser Strategie. KExDaV muss dafür sorgen, dass die entsprechenden Sollparameter von einem System (entsprechende Parameterattributgruppen unter dem Aspekt Soll) in das andere System als Vorgabeparameter (entsprechende Parameterattributgruppen unter dem Aspekt Soll) übertragen werden.

Dazu muss KExDaV sich für die entsprechenden Parameterattributgruppen als Empfänger für den Aspekt Soll beim "Quell"-System anmelden und diese Daten als Sender den "Ziel"-System zur Verfügung stellen. Dabei sind die Angaben der Simulationsvarianten zu berücksichtigen.

⁴ Damit die ausgewählten Parameterdaten in beiden Systemen verwaltet werden, muss natürlich die Parametrierung der jeweiligen Parametrierungsapplikation (Param_L und Param_R) so eingestellt sein, dass sie für die entsprechenden Parameter zuständig ist.

Für diese Strategie ist es sinnvoll, die Vorgabe zum Attribut Delta aus dem Parameterdatensatz zu ignorieren und die Parameterdaten nur bei Änderung übertragen. Damit lassen sich Rückkopplung und ständiges hin und her kopieren nach einer Änderungen unterdrücken.

KEx-DaV muss erkennen, wenn in beiden Systemen fast gleichzeitig unterschiedliche Änderungen an den Parametern durchgeführt werden und ein ständiges hin- und herschalten des Zustands durch Priorisierung des Lokal-Systems unterdrücken.

4.1.4.2.6 Verwaltung der Parameter im Lokal- und Remote-System (Austausch nur bei Triggersignal)

TAnf-KExDaV 19: Verwaltung Parameter Lokal- und Remote-System bei Trigger

Diese Strategie entspricht im Wesentlichen der "Verwaltung der Parameter im Lokal- und Remote-System". Dabei wird der Austausch von Parameterdaten nur nach einem Triggersignal durchgeführt.

4.1.4.2.7 Triggerung für den Parameteraustausch

TAnf-KExDaV 20: TriggerKExDaV

Es ist ein Trigger vorzusehen, über den je KExDaV-Objekt die Parameterdaten zu allen oder zu einzelnen Remotesystemen (spezifizierbar über die PIDs der jeweiligen Remote-DaV) gesteuert werden können. Bei dem Trigger muss die Richtung des Parameterdatenaustausches vorgegeben werden. Dabei wird nach Erhalt des Triggersignals jeweils einmalig der Austausch der Parameterdaten durchgeführt.

Die Folgende Abbildung skizziert den Aufbau der Attributgruppe zur Triggerung für den Parameteraustausch:

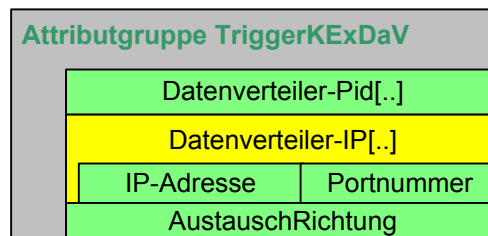


Abbildung 4-11: Attributgruppe TriggerKExDaV

Der Remote-Datenverteiler kann sowohl über die PID des Datenverters als auch über die Vorgabe von IP-Adresse und Portnummer angegeben werden. Über das Attribut AustauschRichtung wird die Richtung des Parameteraustausches vorgegeben (Lokal → Remote bzw. Remote → Lokal).

Der Trigger wirkt sich nur bei der Strategie "Verwaltung der Parameter im Lokal- und Remote-System (Austausch nur bei Triggersignal)" aus.

4.1.4.2.8 Behandlung bei unterschiedlichen Konfigurationsständen für Parameterattributgruppen oder Aspekte

TAnf-KExDaV 21: Unterschiedliche Konfigurationsstände

Für die Parameterattributgruppen und -Aspekte ist analog zu Kapitel 4.1.3.2.1 "Behandlung bei unterschiedlichen Konfigurationsständen für Attributgruppen oder Aspekte" zu verfahren.

4.1.4.2.9 Behandlung von Referenzen auf Objekte in den Parameterdatensätzen

TAnf-KExDaV 22: Referenzen in Parameterdatensätzen

In Parameterdatensätzen können Referenzen auf Objekte enthalten sein. Für diese Fälle muss KExDaV prüfen, ob die Referenz in beiden Systemen auflösbar ist. Dabei ist analog zu Kapitel 4.1.3.2.2 "Behandlung von Referenzen auf Objekte in den Onlinedatensätzen" zu verfahren.

4.1.5 Plugin-Schnittstelle für den Austausch von Datensätzen zu Attributgruppen

TAnf-KExDaV 23: Plugin-Schnittstelle

Bei dem Austausch von Online- oder Parameterdaten besteht die Möglichkeit, im Quell- und Zielsystem unterschiedliche Attributgruppen zu verwenden. Für diese Abbildungen ist eine Plugin-Schnittstelle vorzusehen, über die diese Abbildungen durchgeführt werden.

Ein Plugin muss dabei jeweils angeben, welche Datensätze es von einer in eine andere Attributgruppe umsetzen kann. Die erstellten Plugin werden über Aufrufargumente KExDaV beim Start bekannt gemacht. Wenn laut Parameter ein Datensatz von einer auf eine andere Attributgruppe umgesetzt werden soll, wird ein entsprechendes Plugin gesucht, das diese Umsetzung bietet. Wenn kein entsprechendes Plugin vorhanden ist, ist eine entsprechende Warnung und Betriebsmeldung auszugeben.

4.1.6 Austausch von dynamischen Objekten

TAnf-KExDaV 24: Austausch von dynamischen Objekten

KExDaV muss den Austausch von dynamischen Objekten zwischen dem Lokal-System und dem Remote-System ermöglichen. Dazu muss jedem System einen Konfigurationsbereich zur Verfügung gestellt werden, in dem KExDaV dynamische Objekte anlegen, ändern und löschen kann. Die beiden Bereiche werden über die Parameter `KBLokal` und `KBRemote` referenziert (s. Kapitel 4.1.2 "Start und Initialisierung von KExDaV").

Die Auswahl der auszutauschenden dynamischen Objekte erfolgt über die Vorgabe der Typen, zu denen dynamischen Objekte ausgetauscht werden sollen. Dazu sind folgende Angaben erforderlich bzw. optional möglich:

AustauschDynamischeObjekteLokalNachRemote[..]				
AuswahlBereichTypen[..]			AuswahlRegionTypen[..]	
KV[..]	KB[..]	Typ[..]	Region[..]	Typ[..]
AustauschDynamischeObjekteRemoteNachLokal[..]				
AuswahlBereichTypen[..]			AuswahlRegionTypen[..]	
KV[..]	KB[..]	Typ[..]	Region[..]	Typ[..]

Abbildung 4-12: Parameter Austausch von dynamischen Objekten

- **AustauschDynamischeObjekteLokalNachRemote[0..]**

Attributliste als Array der Größe 0 bis unbegrenzt.

Für diese Einträge werden nur dynamische Objekte vom Lokal-System zum Remote-System übertragen.

- **AuswahlBereichTypen[0..]**

Attributliste als Array der Größe 0 bis unbegrenzt.

Hier kann eine Filterung der auszutauschenden Objekte durch die Vorgabe von Konfigurationsverantwortlichen oder Konfigurationsbereichen erfolgen.

Wenn dieses Array leer ist, erfolgt (hierüber) keine Auswahl von auszutauschenden dynamischen Objekten.

- **Array Konfigurationsverantwortliche [0 ..]**

Über dieses Array können Konfigurationsverantwortliche angegeben werden. Die Auswahl eines Konfigurationsverantwortlichen bedeutet, dass alle Konfigurationsbereiche dieses Konfigurationsverantwortlichen betrachtet werden.

Wenn das Array Konfigurationsverantwortliche leer ist, ist kein Konfigurationsverantwortlicher und damit kein Konfigurationsbereich ausgewählt.

- **Array Konfigurationsbereich [0 ..]**

Über dieses Array können Konfigurationsbereiche angegeben werden. Die Auswahl eines Konfigurationsbereichs bedeutet, dass alle dynamischen Objekte dieses Konfigurationsbereichs betrachtet werden.

Wenn das Array Konfigurationsbereich leer ist, ist kein Konfigurationsbereich ausgewählt.

- **Array Typen [0 ..]**

Über dieses Array werden die bisher auszutauschenden dynamischen Objekte auf die angegebenen Typobjekte beschränkt. Wenn die beiden Arrays Konfigurationsverantwortliche und Konfigurationsbereich beide zu keiner (Vor)Auswahl von Konfiguri-

onsbereichen geführt haben, wird an dieser Stelle die gesamte Konfiguration betrachtet.

Wenn das Array Typen leer ist, erfolgt keine Filterung nach Objekttypen. In diesem Fall sind alle dynamischen Objekte der ausgewählten Konfigurationsbereiche zu behandeln.

- **AuswahlRegionTypen[0..]**

Attributliste als Array der Größe 0 bis unbegrenzt.

Hier kann eine Filterung der auszutauschenden Objekte durch die Vorgabe von definierten Regionen erfolgen. Wenn dieses Array leer ist, erfolgt (hierüber) keine Auswahl von auszutauschenden dynamischen Objekten.

- **Array Region [0 ..]**

Über dieses Array können bereits definierte Regionen in der Konfiguration angegeben werden. Die Auswahl einer Region bedeutet, dass alle dynamischen Objekte dieser Region betrachtet werden.

Wenn das Array Region leer ist, ist alle dynamischen Objekte ausgewählt.

- **Array Typen [0 ..]**

Über dieses Array werden die bisher auszutauschenden dynamischen Objekte auf die angegebenen Typobjekte beschränkt. Wenn die beiden Arrays Konfigurationsverantwortliche und Konfigurationsbereich beide zu keiner (Vor)Auswahl von Konfigurationsbereichen geführt haben, wird an dieser Stelle die gesamte Konfiguration betrachtet.

Wenn das Array Typen leer ist, erfolgt keine Filterung nach Objekttypen. In diesem Fall sind alle dynamischen Objekte der ausgewählten Konfigurationsbereiche zu behandeln.

- **AustauschDynamischeObjekteRemoteNachLokal[0..]**

Attributliste als Array der Größe 0 bis unbegrenzt.

Für diese Einträge werden nur dynamische Objekte vom Remote-System zum Lokal-System übertragen.

Der weitere Aufbau entspricht der Liste AustauschDynamischeObjekteLokalNachRemote.

Abbildung 4-13 skizziert den Ablauf beim Austausch von dynamischen Objekten. In dem Beispiel werden für das Remote-System die dynamischen Objekte aus den 3 Konfigurationsbereiche $KB1_L$, $KB2_L$ und $KB3_L$ der Konfiguration $Konf_L$ in der Richtung Lokal→Remote-System ausgetauscht. KExDaV speichert die entsprechenden dynamischen Objekte im Konfigurationsbereich KB_{Lokal} , der von der Konfiguration $Konf_R$ geladen wird.

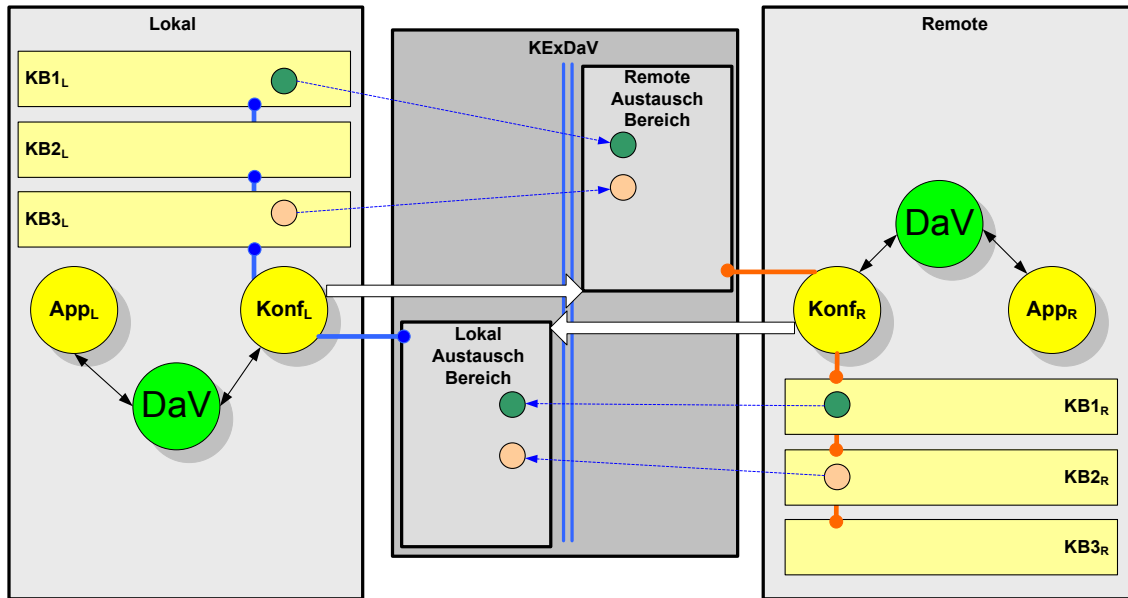


Abbildung 4-13: Ablauf Austausch von dynamischen Objekten

Hinweis: Der Remote Austausch Bereich und der Lokal Austauschbereich kann jeweils aus mehreren Konfigurationsbereichen bestehen s. Kapitel 4.1.2 "Start und Initialisierung von KExDaV". Aus Gründen der Übersichtlichkeit ist in Abbildung 4-13: Ablauf Austausch von dynamischen Objekten und im weiteren Textverlauf jeweils nur ein Bereich aufgeführt.

4.1.6.1 Initialisierung

TAnf-KExDaV 25: Initialisierung

Beim Start von KExDaV müssen die Konfigurationsbereiche `RemoteAustauschBereich` und `LokalAustauschBereich` initialisiert werden. Hierbei sind drei verschiedene Ausgangspunkte möglich⁵:

- Der referenzierte Bereich `RemoteAustauschBereich` ist nicht vorhanden bzw. undefiniert. In diesem Fall ist der Austausch von dynamischen Objekten in der Richtung Lokal→Remote-System nicht möglich. Falls laut Parametrierung diese Richtung vorgegeben ist, handelt es sich um einen schwerwiegenden Fehler, und KExDaV muss mit entsprechender Fehlermeldung beendet werden. Falls laut Parametrierung diese Richtung nicht vorgegeben ist, wird KExDaV weiter betrieben und eine Warnung ausgegeben. Wenn im laufenden Betrieb der Parameter so geändert wird, dass der Bereich erforderlich

⁵ Im Folgenden wird die Initialisierung für den Konfigurationsbereich `RemoteAustauschBereich` betrachtet. Die Initialisierung des Konfigurationsbereichs `LokalAustauschBereich` erfolgt analog.

wird, muss KExDaV den Parametersatz ignorieren und eine entsprechende Betriebsmeldung und Debugausgabe erzeugen. Dabei ist der ungeänderte Parameter erneut unter dem Aspekt "Ist" zu publizieren.

- Der referenzierte Bereich `RemoteAustauschBereich` ist leer
- Der referenzierte Bereich `RemoteAustauschBereich` enthält bereits dynamische Objekte

Wenn der Bereich nicht leer ist, muss jedes enthaltene dynamische Objekt in Bezug auf Existenz eines korrespondierenden Objektes verifiziert werden. Das bedeutet, dass die dynamischen Objekte, die nicht mehr im Quell-System (hier Lokal-System) vorhanden sind, gelöscht werden müssen. Diese Prüfung kann parallel zur Erzeugung der dynamischen Objekte erfolgen.

Vorgehensweise

KExDaV kontrolliert mit Hilfe der Konfiguration welche dynamischen Objekte gemäß Parametrierung im Quell-System vorhanden und zu berücksichtigen sind.

Dazu meldet sich KExDaV auf das Erzeugen und Löschen von entsprechenden Objekten an.

Es werden initial alle dynamischen Objekte ermittelt, die übertragen werden sollen. Je dynamisches Objekt wird geprüft,

- ob bereits ein entsprechendes Objekt in dem Zielsystem vorhanden ist. Wenn dies der Fall ist, wird geprüft, ob sich die konfigurierenden Daten zu dem dynamischen Objekt geändert haben. Falls dies der Fall ist, werden die entsprechenden konfigurierenden Daten im Bereich `RemoteAustauschBereich` angepasst. Das bedeutet, dass alte Objekt wird gelöscht und ein neues Objekt angelegt.
- wenn noch kein entsprechendes Objekt in dem Zielsystem vorhanden wird ein entsprechendes dynamisches Objekt im Bereich `RemoteAustauschBereich` angelegt (s. Kapitel 4.1.6.4 "Kopieren eines dynamischen Objektes").
- Nachdem alle ermittelten dynamischen Objekte in dieser Form geprüft wurden, werden die eventuell schon vorhandenen Objekte, die im Quellsystem nicht mehr gültig sind im Bereich `RemoteAustauschBereich` gelöscht (s. Kapitel 4.1.6.5 "Löschen eines dynamischen Objektes").

4.1.6.2 Laufender Betrieb

TAnf-KExDaV 26: Laufender Betrieb

Im laufenden Betrieb reagiert KExDaV entsprechend.

- Es wird ein neues zu berücksichtigendes dynamisches Objekt im Quellsystem angelegt → Kapitel 4.1.6.4 "Kopieren eines dynamischen Objektes".
- Ein zu berücksichtigendes dynamischen Objektes wird im Quellsystem gelöscht → Kapitel 4.1.6.5 "Löschen eines dynamischen Objektes".

4.1.6.3 Parameteränderung

TAnf-KExDaV 27: Parameteränderungen

Eine Parameteränderung muss sich sofort auswirken⁶.

4.1.6.4 Kopieren eines dynamischen Objektes

TAnf-KExDaV 28: Kopieren dynamische Objekte

Beim Kopieren wird ein neues dynamischen Objektes entsprechenden Typs im Bereich `RemoteAustauschBereich` angelegt. Dabei muss KExDaV alle konfigurierenden Datensätze des Quellobjekts für das neu zu erzeugende Zielobjekt übernehmen.

In den konfigurierenden Datensätzen können Referenzen auf Objekte enthalten sein. Für diese Fälle muss KExDaV die korrespondierenden Objekte ermitteln (s. Kapitel 4.1.3.1.1 "Ermittlung korrespondierender Konfigurationsobjekte").

Referenz auf ein Konfigurationsobjekt

Wenn im Zielsystem ein korrespondierendes Objekt vorhanden ist, ist eine Referenz auf dieses Konfigurationsobjekt in dem konfigurierenden Datensatz einzutragen.

Wenn das nicht der Fall ist, muss geprüft werden, ob das Attribut den Wert undefiniert zulässt. Falls dies der Fall ist, ist an der entsprechenden Stelle undefiniert einzutragen.

Falls der Wert undefiniert nicht zugelassen ist, kann der konfigurierende Datensatz nicht geschrieben werden. In diesem Fall wird der entsprechende konfigurierende Datensatz unterdrückt und eine entsprechende Meldung ausgegeben (Debug, Betriebsmeldung).

Referenz auf ein dynamisches Objekt

Wenn im Zielsystem ein korrespondierendes Objekt vorhanden ist, ist eine Referenz auf dieses Konfigurationsobjekt in in dem konfigurierenden Datensatz einzutragen.

Wenn das nicht der Fall ist, muss KExDaV dafür sorgen, dass das entsprechende dynamische Objekt in dem Zielsystem entsprechend angelegt wird. Dies erfolgt analog zu Kapitel 4.1.6 "Austausch von dynamischen Objekten".

4.1.6.5 Löschen eines dynamischen Objektes

TAnf-KExDaV 29: Löschen dynamische Objekte

Beim Löschen wird das entsprechende dynamische Objekt im Bereich `RemoteAustauschBereich` gelöscht bzw. auf ungültig gesetzt.

⁶ Dabei ist zu beachten, dass vom Parameter referenzierte Objekte ebenfalls änderbar sind. Z.B. ist parametrierbar, welche Objekte zu einer Region gehören.

4.1.7 Austausch von dynamischen Mengen

TAnf-KExDaV 30: Austausch von dynamischen Mengen

KExDaV muss den Austausch von dynamischen Mengen zwischen dem Lokal-System und dem Remote-System ermöglichen.

Bei dynamischen Mengen sind folgende Dinge zu beachten:

- Die Systeme Lokal-System und Remote-System müssen jeweils unter eigenen Konfigurationsverantwortlichen betrieben werden.
- Eine dynamische Menge hängt unter einem Konfigurationsobjekt, und darf daher nur von einer Konfiguration bearbeitet werden, nämlich der, die für das entsprechend Konfigurationsobjekt bzw. dem Bereich, dem das Konfigurationsobjekt zugehörig, verantwortlich ist. Diese Verantwortlichkeit wird durch den Konfigurationsverantwortlichen modelliert.
- Da Konfigurationsobjekte im Gegensatz zu dynamischen Objekten nicht automatisch über KExDaV zwischen dem Lokal-System und dem Remote-System ausgetauscht werden können, muss für den Austausch von dynamischen Mengen in beiden Systemen jeweils ein konkretes Konfigurationsobjekt vorhanden sein.

Die aufgeführten Punkte verdeutlichen, dass der Austausch von dynamischen Mengen kompliziert ist und dass bei der Parametrierung zum Austausch von dynamischen Mengen besondere Sorgfalt erforderlich ist.

Zur Verdeutlichung sollen an dieser Stelle mögliche Anwendungsfälle skizziert werden. Dazu werden folgende dynamische Mengen betrachtet:

- **Meldungen** (PID: `menge.meldungen`)
Diese dynamische Menge kann für Konfigurationsobjekte vom Typ `BetriebsmeldungsVerwaltung` (`typ.betriebsMeldungsVerwaltung`) verwendet werden. Sie verwaltet dynamische Objekte vom Typ `Meldung` (`typ.meldung`).
- **Staus** (PID: `menge.staus`)
Diese dynamische Menge muss für Konfigurationsobjekte vom Typ `VerkehrsModellNetz` (`typ.verkehrsModellNetz`) verwendet werden. Sie verwaltet dynamische Objekte vom Typ `Stau` (`typ.stau`).

4.1.7.1 Anwendungsfall Meldungen

Jedes selbständig lauffähige datenverteilerbasierte System wird unter einer so genannten Autarken Organisationseinheit betrieben. Die autarke Organisationseinheit vereinigt wichtige Eigenschaften des Systems. Über ein Konfigurationsobjekt dieses Typs erfolgt quasi eine Zusammenfassung von einem Konfigurationsverantwortlichen, einer Parametrierung und einer Betriebsmeldungsverwaltung sowie weiterer systembezogener Objekte. Modelltechnisch wird das so umgesetzt, indem ein Objekt vom Typ `AutarkeOrganisationsEinheit` andere Typen erweitert. In der folgenden Auflistung sind einige dieser Typen aufgeführt.

- Archiv
- `BetriebsMeldungsVerwaltung`
- Kalender
- `KonfigurationsVerantwortlicher`
- Parametrierung

- etc.

Der kleine Exkurs sollte zeigen, dass die Menge Meldungen auch an einem Objekt vom Typ `AutarkeOrganisationsEinheit` verwaltet werden kann. Der Meldungsverwaltung wird per Aufrufargument übergeben, bei welchem Konfigurationsobjekt die zu verwaltende Menge der Meldungen hängt.

Als Anwendungsfall soll das Remote-System die Meldungen erhalten, die die Betriebsmeldungsverwaltung des Lokal-Systems verwaltet. Dabei wird davon ausgegangen, dass die Meldungen an dem Konfigurationsobjekt hängen, das die Autarke Organisationseinheit des Lokal-Systems darstellt (Konfigurationsobjekt `AOELokal`).

Im Folgenden werden zwei Varianten skizziert, wie der Austausch dieser Menge erfolgen kann.

Variante I

Abbildung 4-14 zeigt den Fall, dass die Menge der Meldungen vom Lokal-System auf die Autarke Organisationseinheit des Remote-Systems (Konfigurationsobjekt `AOERemote`) abgebildet wird. Also:

`AOELokal:Meldungen` → `AOERemote:Meldungen`

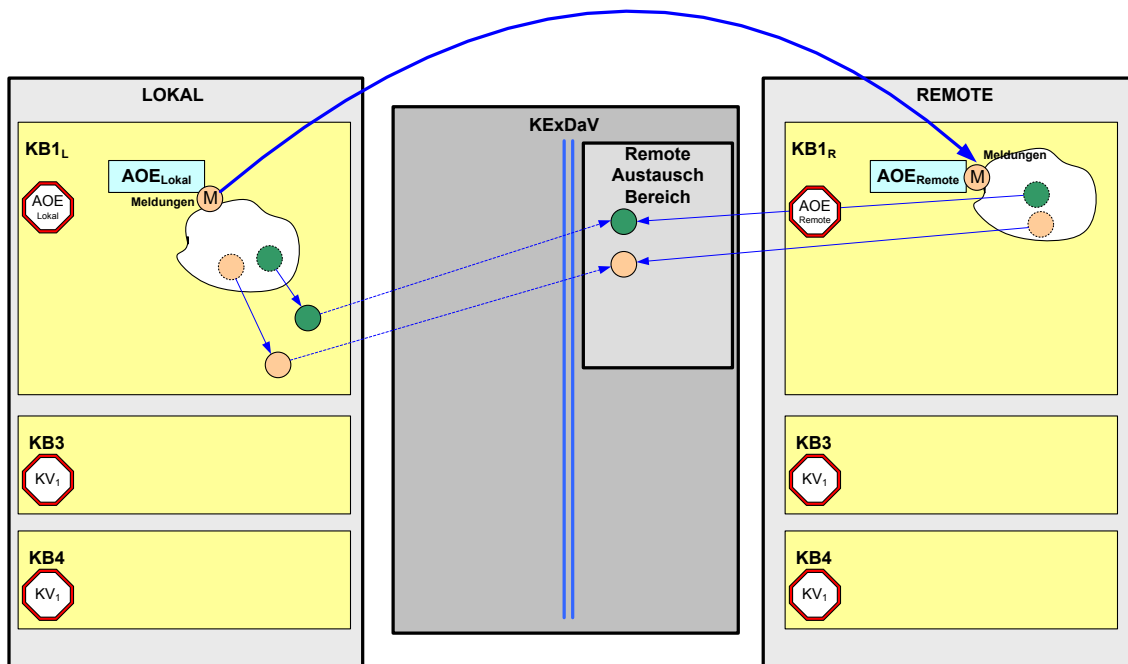


Abbildung 4-14: Austausch dynamische Menge Meldungen Variante I

Bei dieser Variante könnte bei dem Remote-System auf eine eigene Betriebsmeldungsverwaltung verzichtet werden. Die Meldungen vom Lokal-System werden in die Menge Meldungen übertragen, wo im Normalfall die Betriebsmeldungen verwaltet werden. Damit auch Meldungen aus dem Remote-System in der Menge verwaltet werden, müsste dafür gesorgt werden, dass der entsprechende Datenfluss der Onlinedaten zu den Informationskanälen über den Austausch von Onlinedaten "umgebogen" wird.

Abbildung 4-15 zeigt das Arbeitsschema der Betriebsmeldungsverwaltung. Die Betriebsmeldungsverwaltung meldet sich als Senke für Informationsmeldungen an und publiziert diese gefiltert wieder bzw. legt nach parametrisierten Regeln Meldungsobjekte an und verwaltet diese.

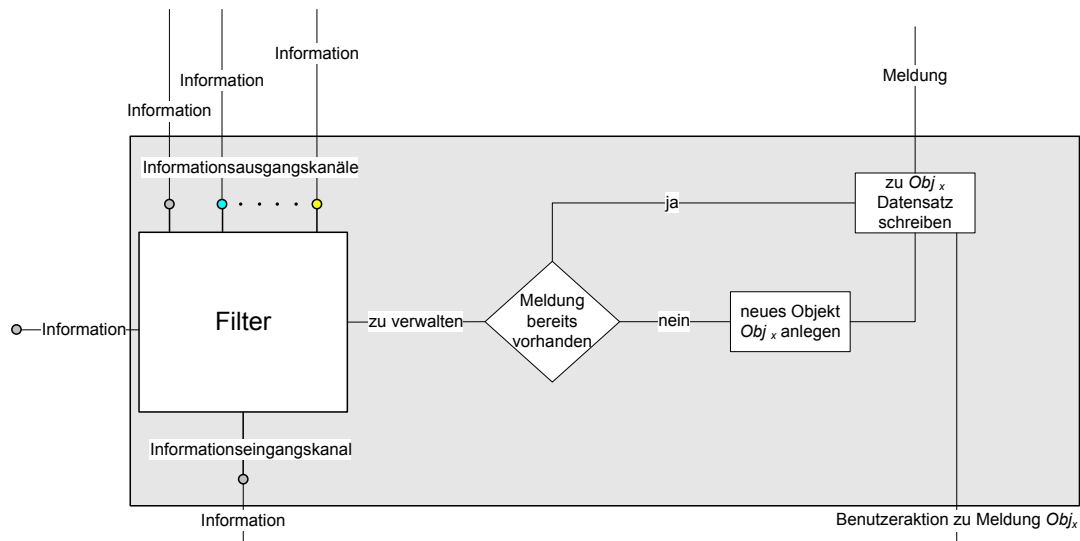


Abbildung 4-15: Schema Betriebsmeldungsverwaltung

Die im Remote-System entstehenden Informationsmeldungen müssten also über den Austausch von Onlinedaten an die Betriebsmeldungsverwaltung des Lokal-Systems übertragen werden.

Variante II

Abbildung 4-16 zeigt den Fall, dass die Menge der Meldungen vom Lokal-System auf ein Konfigurationsobjekt vom Typ `Betriebsmeldungsverwaltung` im Remote-System abgebildet wird (Konfigurationsobjekt `BMV`). Also

`AOELokal:Meldungen` → `BMV:Meldungen`

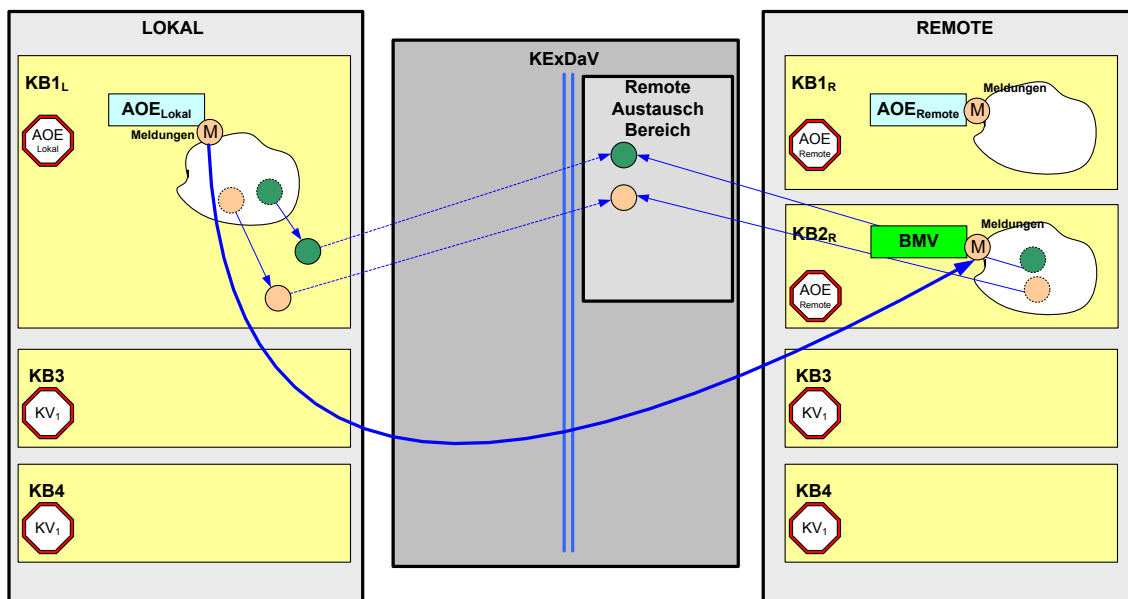


Abbildung 4-16: Austausch dynamische Menge Meldungen Variante II

In dieser Variante könnte das Remote-System eine eigene unabhängige Betriebsmeldungsverwaltung haben und zusätzlich auf die Meldungen vom Lokal-System über das hierfür extra konfigurierte Konfigurationsobjekt `BMV` zugreifen.

4.1.7.2 Anwendungsfall Staus

Als weiterer Anwendungsfall soll die Menge von Staus betrachtet werden, die vom Lokal-System auf das Remote-System übertragen werden sollen.

Konfigurationsobjekte vom Typ `VerkehrsModellNetz` (`typ.verkehrsModellNetz`) müssen eine dynamische Menge `Staus` vom Typ `Stau` (`typ.stau`) verwalten. Damit die dynamische Menge übertragen werden kann, muss auf dem Remote-System ein Konfigurationsobjekt zur Verfügung gestellt werden, das unter dem AOE `AOERemote` bearbeitet werden darf. Hier ist folgende Abbildung erforderlich:

`VMNetzLokal:Staus` → `VMNetzRemote:Staus`

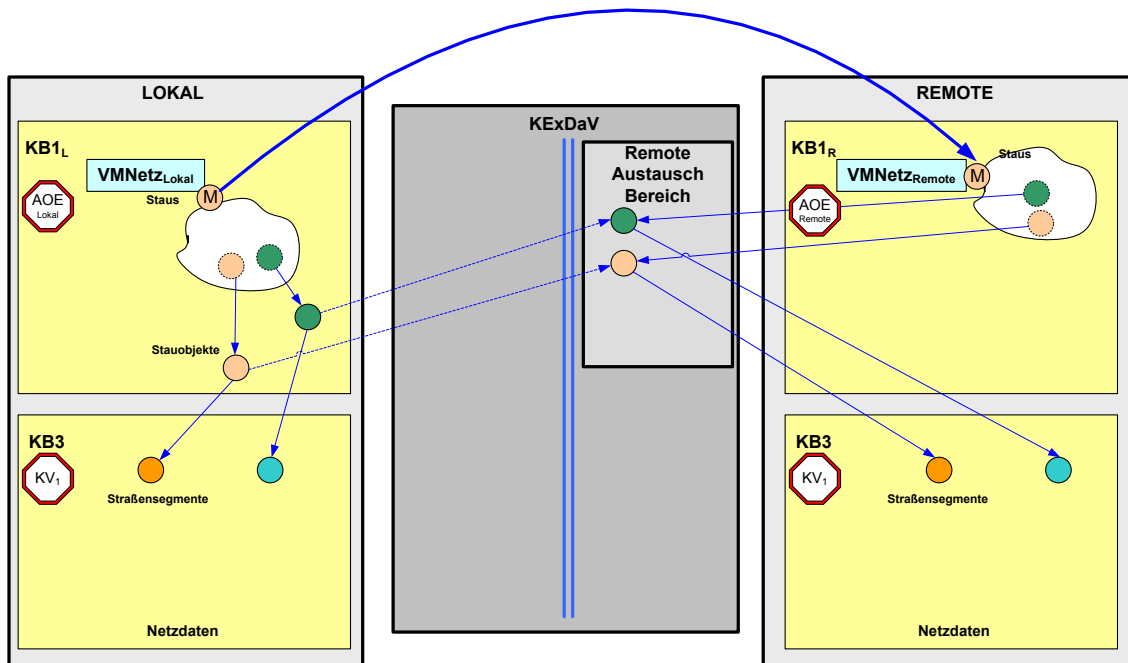


Abbildung 4-17: Austausch dynamische Menge Staus

Abbildung 4-17 skizziert, was für diesen Fall zu beachten ist:

- Für das Remote-System muss das Konfigurationsobjekt `VMNetzRemote` konfiguriert sein und durch den Konfigurationsverantwortlichen (bzw. der AOE) des Remote-Systems muss die zugeordnete Menge `Staus` geändert werden können.
- Vom Remote-System muss der Konfigurationsbereich `RemoteAustauschBereich` zur Verfügung gestellt werden, in den KExDaV die entsprechenden dynamischen Stauobjekte übertragen kann (s. Kapitel 4.1.6 "Austausch von dynamischen Objekten").
- Die Stauobjekte referenzieren Konfigurationsobjekte vom Typ `StraßenSegment`. Der oder die erforderlichen Konfigurationsbereiche müssen in der Konfiguration des Remote-Systems eingebunden sein.

4.1.7.3 Spezifikation Austausch von dynamischen Mengen

TAnf-KExDaV 31: Spezifikation Austausch

Die Auswahl der auszutauschenden dynamischen Mengen erfolgt über die Vorgabe der Mengen, zu denen dynamischen Objekte ausgetauscht werden sollen. Dazu sind folgende Angaben erforderlich bzw. optional möglich:

AustauschDynamischeMengen[..]				
DynamischeMengeLokal		DynamischeMengeRemote		Richtung
Objekt	Menge	Objekt	Menge	

Abbildung 4-18: Parameter Austausch von dynamischen Mengen

- **AustauschDynamischeMengen**

Attributliste als Array der Größe 0 bis unbegrenzt.

- **DynamischeMengeLokal**

Attributliste

Hier wird die dynamischen Menge, die im Lokal-System betrachtet wird, spezifiziert.

- **Objekt**

Konfigurationsobjekt, an dem die betrachtete Menge verwaltet wird.

- **Menge**

Name der betrachteten Menge.

- **DynamischeMengeRemote**

Attributliste

Hier wird die dynamischen Menge, die im Remote-System betrachtet wird, spezifiziert.

- **Objekt**

Konfigurationsobjekt, an dem die betrachtete Menge verwaltet wird.

- **Menge**

Name der betrachteten Menge.

- **Richtung**

Über dieses Attribut wird die Richtung, in der die dynamischen Mengen ausgetauscht werden sollen festgelegt. Folgende Angaben sind möglich:

- **Lokal→Remote-System**

In diesem Fall wird die dynamische Menge vom Lokal-System zum Remote-System übertragen.

- **Remote→Lokal-System**

In diesem Fall wird die dynamische Menge vom Remote-System zum Lokal-System übertragen.

4.1.7.4 Initialisierung

TAnf-KExDaV 32: Initialisierung

KExDaV muss initial für jede dynamische Menge den Anfangszustand bestimmen.

Vorgehensweise:

KExDaV liest die jeweilige dynamische Menge des Quellsystems (im Folgenden auch Quellmenge genannt) ein und prüft, für jedes Element dieser Menge, ob in der dynamischen Menge des Zielsystems (im Folgenden auch Zielmenge genannt) bereits ein entsprechendes Objekt enthalten ist.

- Wenn noch kein entsprechendes Objekt enthalten ist, wird geprüft, ob im Zielsystem ein entsprechendes Objekt vorhanden ist, das zu der Menge hinzugefügt werden kann.
 - Wenn dies der Fall ist, wird das entsprechende Objekt der Zielmenge hinzugefügt.
- Wenn kein entsprechendes Objekt vorhanden ist, wird unterschieden ob es sich um ein Konfigurationsobjekt oder ein dynamisches Objekt handelt, das der Zielmenge hinzugefügt werden soll.
 - Wenn es sich um ein dynamisches Objekt handelt, versucht KExDaV das entsprechende dynamische Objekt vom Quellsystem zu übertragen s.a. Kapitel 4.1.6 "Austausch von dynamischen Objekten".
 - Wenn es sich um ein Konfigurationsobjekt handelt oder kein entsprechendes dynamisches Objekt aus dem Quellsystem übertragen werden kann, wird dieses Element unterdrückt und damit nicht zur Zielmenge zugefügt.
- Nachdem alle Objekte der Quellmenge abgearbeitet wurden, wird geprüft, ob in der Zielmenge noch Objekte vorhanden sind, die kein entsprechendes Element der Quellmenge aufweisen. Diese Objekte werden aus der Zielmenge entfernt.

4.1.7.5 Laufender Betrieb

TAnf-KExDaV 33: Laufender Betrieb

Im laufenden Betrieb werden die Quellmengen bzgl. Änderungen überwacht und es werden die erkannten Änderungen an der Zielmenge durchgeführt.

4.1.7.6 Parameteränderung

TAnf-KExDaV 34: Parameteränderung

Eine Parameteränderung muss sich sofort auswirken⁷.

⁷ Dabei ist zu beachten, dass vom Parameter referenzierte Objekte ebenfalls änderbar sind. Z.B. ist parametrierbar, welche Objekte zu einer Region gehören.

4.2 Technische Anforderungen an die Schnittstellen

Die SW-Einheit besitzt folgende Schnittstellen

- KExDaV – Starter
(Aufrufschnittstelle der Applikation)
- Datenverteiler Applikationsfunktionen – Applikation
(Schnittstelle zur SW-Einheit „Datenverteiler-Applikationsfunktionen“ s. [TAnfDaV])
- KExDaV – Applikation
(logische Schnittstelle zu anderen Applikationen)

4.2.1 Technische Anforderungen an die Nutzerschnittstelle

4.2.1.1 Schnittstelle KExDaV – Starter

TAnf-KExDaV 35: Schnittstelle KExDaV – Starter

Die SW-Einheit stellt im Sinne der Festlegungen der Systemarchitektur [SysArc] eine Applikation dar und verfügt damit über mehrere benannte Aufrufparameter, mit denen sich das (Start-) Verhalten der Applikation in bestimmten Bereichen einstellen lässt. Folgende Aufrufparameter sind zu unterstützen:

`-kexDaV=PidDesKexDaV`

Pro Datenverteilerbasiertem System können mehrere SW-Einheiten KExDaV eingesetzt werden. Über diesen Parameter wird der KExDaV-Applikation mitgeteilt, an welchem Objekt der Parameter zur Spezifikation des KExDaV "hängt"

`-plugin=KlasseDesPlugin`

KExDaV unterstützt die Abbildung von Datensätzen von einer in eine andere Attributgruppe. Über diesen Parameter werden die entsprechenden Plugins beim Start der KExDaV-Applikation bekannt gemacht.

4.2.2 Technische Anforderungen an andere Schnittstellen

4.2.2.1 Schnittstelle KExDaV – Starter

Siehe Kapitel 4.2.1.1 "Schnittstelle KExDaV – Starter".

4.2.2.2 Schnittstelle KExDaV – Applikation

TAnf-KExDaV 36: Schnittstelle KExDaV – Applikation

Logische Schnittstelle, die unter Nutzung der Datenverteilerschnittstelle realisiert ist und durch die Beschreibung der ausgetauschten Daten und eventuell notwendiger Abläufe beim Datenaustausch spezifiziert wird.

4.3 Qualitätsforderungen

4.3.1 Kritikalität

Entsprechend der projektspezifischen Kritikalitätsdefinitionen wird die Kritikalität dieses Segments als **mittel** eingestuft, da Fehlverhalten zu einer informationstechnischen Fehlfunktion (erkannter Fehler, z. B. Protokollfehler) an einer externen Systemschnittstelle führt oder zu einer Einschränkung der Datenerfassung der an den externen Schnittstellen erfassten Daten (z. B. fehlerhafte Archivierung).

4.3.2 Technische Anforderungen der IT-Sicherheit

Es werden keine über die IT-Sicherheitsanforderungen der [Afo] hinausgehenden Anforderungen an das Segment gestellt.

4.3.3 Technische Anforderungen an sonstige Qualitätsmerkmale

Für dieses Segment gelten die allgemeinen Anforderungen an die Anwendersoftware, wie in der [Afo] Kapitel 6.7 "Qualitätsanforderungen" festgelegt.

4.3.4 Technische Anforderungen an die Entwicklungs- und SWPÄ-Umgebung

Für dieses Segment gelten die allgemeinen Anforderungen an die Anwendersoftware, wie in der [Afo] Kapitel 7.1.1 "Anforderungen an die Systemsoftware" festgelegt.

5 SW Architektur der SW-Einheit "KExDaV"

5.1 Lösungsvorschläge

5.1.1 Kriterien für die Zerlegung der SW-Einheit

Hauptkriterien für die Zerlegung der SW-Einheit in Module sind:

- **Wiederverwendbarkeit** von Modulen
Wenn absehbar ist, dass bestimmte Funktionen der SW-Einheit auch an anderen Stellen im System benötigt werden, dann führt eine entsprechende Modularisierung dazu, dass die jeweiligen Funktionen einfach durch Wiederverwendung der entsprechenden Module in andere SW-Einheiten übernommen werden können.
Andererseits wird die Architektur natürlich durch die Wiederverwendung von Modulen aus anderen Stellen des Systems oder durch die Verwendung von verfügbaren kommerziellen oder Open-Source basierten Fremdmodulen beeinflusst.
- **Austauschbarkeit** von Modulen
Die Austauschbarkeit von Modulen innerhalb einer SW-Einheit ist wichtig, um eine Flexibilität bezüglich der Implementierung bestimmter Funktionen zu erreichen.
Mit dem Austausch eines Moduls durch ein anderes Modul mit gleichen Schnittstellen kann zum einen bei gleicher Funktion des Moduls die Art der Implementierung einfach verändert werden und zum anderen die Funktion des Moduls flexibel verändert werden.
Die Flexibilität kann so weit gehen, dass erst zur Laufzeit der Applikation mit Hilfe von Konfigurationsdaten, Aufrufargumenten oder Parametern entschieden wird, welche Module zum Einsatz kommen.
- **Erweiterbarkeit** der SW-Einheit durch weitere Module mit gleicher Schnittstelle
Die Erweiterbarkeit einer SW-Einheit um weitere Modulen kann ähnlich wie bei der Austauschbarkeit von Modulen über eine entsprechende Schnittstelle vorgesehen werden. Damit erhält die SW-Einheit die notwendige Flexibilität, um zu späteren Zeitpunkten durch weitere Module mit anderen Funktionen ergänzt zu werden.
- **Geringe Schnittstellenkomplexität**
Eine geringe Komplexität der Schnittstellen zwischen den Modulen ist erforderlich, um die Wiederverwendung zu fördern und die Wartbarkeit und Nutzbarkeit der jeweiligen Schnittstellen zu erhöhen.
Folgende Ziele, die zu einer geringen Schnittstellenkomplexität beitragen, sind wesentliche Kriterien für die Modularisierung:
 - Hoher Grad an Abgeschlossenheit der Module
 - Schwache Kopplung der Module untereinander
 - Enge Bindung innerhalb eines Moduls

Wenn mehrere Module gemeinsam einen Funktionskreis implementieren, werden sie zu Komponenten zusammengefasst.

Die sich aus dieser Vorgehensweise herauskristallisierte Modularisierung wird in den folgenden Unterpunkten dargestellt.

5.1.2 Skizzierung der gewählten Lösung

Abbildung 5-1 zeigt die Zerlegung der SW-Einheit in SW-Komponenten und SW-Module sowie die interne Kommunikationsstruktur der SW-Einheit, die sich unter Berücksichtigung der oben erläuterten Kriterien ergibt:

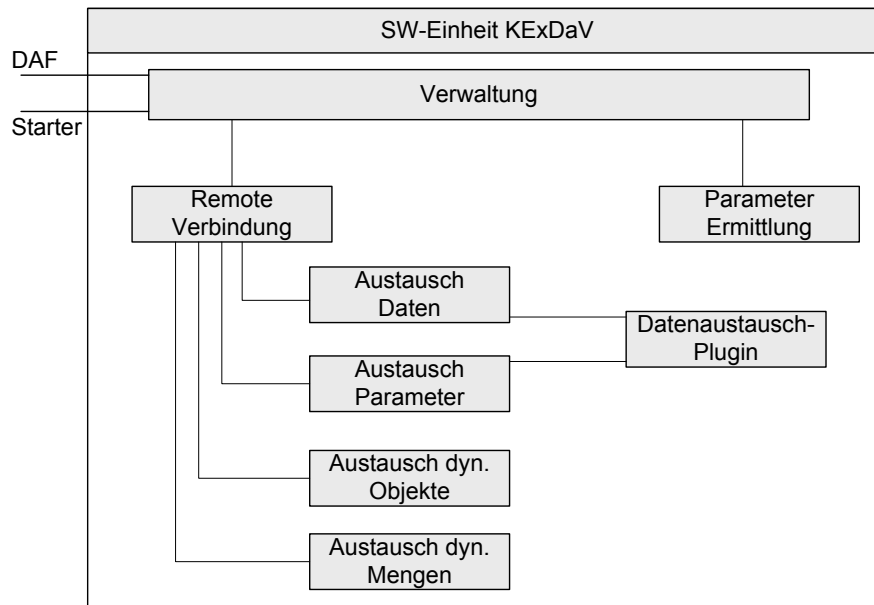


Abbildung 5-1: Zerlegung der SW-Einheit KExDaV

5.2 Modularisierung/Datenbankentwurf

5.2.1 Übersicht der SW-Komponenten, SW-Module, Prozesse und Datenbanken

Die SW-Einheit KExDaV setzt sich aus den folgenden Modulen zusammen:

- Modul Verwaltung
- Modul Remote Verbindung
- Modul Parameter Ermittlung
- Modul Austausch Daten
- Modul Austausch Parameter
- Modul Datenaustausch-Plugin
- Modul Austausch dynamische Objekte
- Modul Austausch dynamische Mengen

In Tabelle 5-1 sind die Komponenten und Module mit ihren eindeutigen Identifikatoren dargestellt:

Identifikator	Langbezeichnung
SE-KExDaV.01	Modul Verwaltung
SE-KExDaV.02	Remote Verbindung

Identifikator	Langbezeichnung
SE-KExDaV.03	Parameter Ermittlung
SE-KExDaV.04	Austausch Daten
SE-KExDaV.05	Austausch Parameter
SE-KExDaV.06	Datenaustausch-Plugin
SE-KExDaV.07	Austausch dynamische Objekte
SE-KExDaV.08	Austausch dynamische Mengen

Tabelle 5-1: Identifizierung der SW-Komponenten und Module der SW-Einheit KExDaV

5.2.2 Einzelbeschreibung

5.2.2.1 Modul Verwaltung

Das Modul Verwaltung wertet die Aufrufargumente aus und sorgt für die Verbindung mit dem lokalen Datenverteiler. Es folgt die Ermittlung des KExDaV-Systemobjekts aus der Konfiguration und für das Starten der Parameter-Ermittlung.

Beim Eintreffen von Parametern sorgt das Modul Verwaltung dafür, dass die entsprechenden Verbindungen zu den Remote-Datenverteilern aufgebaut werden und die Parameter weitergeleitet werden. Diese Aufgaben werden durch das Modul Remote-Verbindung übernommen.

Daneben bietet das Modul Verwaltung eine Schnittstelle, um von allen anderen Modulen Fehlermeldungen, Warnungen etc. entgegenzunehmen und beispielsweise über Debug-Ausgaben oder (je nach Schwere des Fehlers) Betriebsmeldungen zu veröffentlichen. Diese Schnittstelle ist flexibel aufgebaut und kann in Zukunft zusätzlich verwendet werden um beispielsweise Statistiken, Betriebsinformationen und dergleichen zu sammeln und z. B. anzuzeigen.

5.2.2.2 Modul Parameter-Ermittlung

Das Modul Parameter-Ermittlung ermittelt den Parameter am KExDaV-Systemobjekt (s.a. Abbildung 4-2: Parameterattributgruppe SpezifikationKExDaV) und gibt diesen an das Modul Verwaltung weiter. Zusätzlich veröffentlicht es die aktuellen Parameter unter dem Ist-Zustand und meldet sich als Senke auf die Trigger-Attributgruppe zum Triggern von beidseitigem Parametertausch an (s.a. Abbildung 4-11: Attributgruppe TriggerKExDaV).

5.2.2.3 Modul Remote-Verbindung

Das Modul Remote-Verbindung instantiiert pro Remote-Datenverteiler ein Objekt, welches periodisch versucht sich mit dem spezifizierten Remote-Datenverteiler zu verbinden. Sobald die Verbindung zustande gekommen ist, wird anhand der Parameter der Austausch von Daten, Parameterdaten, Objekten und Mengen gestartet.

Zusätzlich verwaltet dieses Modul die mit dem fremden System ausgetauschten dynamischen Objekte, die aus verschiedenen Quellen stammen können:

- Parametrierter Austausch dynamischer Objekte
- Parametrierter Austausch dynamischer Mengen
- Übertragung wegen einer Referenzierung beim Austausch von Daten oder Parametern

- Übertragung wegen einer Referenzierung in den Konfigurationsdaten eines anderen übertragenen dynamischen Objektes.

Diese Objekte werden durch die Klasse KExDaV-Objekt gekapselt, die verschiedene vereinfachende Funktionen auf ein Systemobjekt bietet:

- Abfrage auf Existenz und Benachrichtigung beim Erstellen/Löschen
- Das Senden und Empfangen von Daten
- Das Löschen und Erstellen von Objekten
- Das Auslesen und Setzen von Konfigurationsdaten

5.2.2.4 Modul Austausch Daten

Dieses Modul ist für den Austausch von Daten zwischen korrespondierenden Objekten auf verschiedenen Datenverteilern zuständig.

5.2.2.5 Modul Austausch Parameter

Dieses Modul ist für den Austausch von Parameter-Daten zwischen korrespondierenden Objekten auf verschiedenen Datenverteilern zuständig.

5.2.2.6 Modul Datenaustausch-Plugin

Die Module "Austausch Daten" und "Austausch Parameter" empfangen Daten von einem Objekt auf einem Datenverteiler und senden diese an das gleiche oder ein anderes Objekt, auf einem anderen Datenverteiler. Dabei können andere Attributgruppen und Aspekte verwendet werden.

Dies geschieht unter Zuhilfenahme eines Plugins, das die Daten modifiziert oder konvertiert. Dabei müssen beim Parameterdatenaustausch Rückkopplung und ständiges hin und her kopieren nach einer Änderungen unterdrückt werden (TAnfKExDaV 18).

Das Datenaustausch-Plugin übernimmt dabei die Aufgabe, die Daten von einem Data-Objekt des einen Datenvertailers in ein Data-Objekt des anderen Datenvertailers zu kopieren bzw. zu konvertieren. Die Standard-Implementierung, die nur die einzelnen Attribute eines Data-Objekts kopiert, wird automatisch für Parameterdaten und Datenübertragungen derselben Attributgruppe sowie für das Kopieren von Konfigurationsdaten genutzt.

5.2.2.7 Modul Austausch dynamische Objekte

Das Modul "Austausch dynamische Objekte" tauscht anhand der Parameter die angegebenen dynamischen Objekte zwischen zwei Datenverteilern aus. Dazu wird die gemeinsame, beim Modul Remote-Verbindung geschilderte Funktionalität benutzt.

5.2.2.8 Modul Austausch dynamische Mengen

Das Modul Austausch dynamische Objekte tauscht Anhand der Parameter die angegebenen dynamischen Mengen und deren Objekten zwischen Datenverteilern aus. Zum Austausch der Objekte wird die gemeinsame, beim Modul Remote-Verbindung geschilderte, Funktionalität benutzt.

5.2.3 Dynamisches Ablaufmodell

Bei dieser SW-Einheit sind keine Vorgaben zum dynamischen Ablaufmodell notwendig.

5.2.4 Kritikalität der SW-Komponenten/SW-Module/Prozesse/Datenbanken

Die Kritikalität der einzelnen SW-Komponenten und Module ist in Tabelle 5-2 aufgeführt:

Identifikator	Langbezeichnung	Kritikalität
SE-KExDaV.01	Modul Verwaltung	Mittel
SE-KExDaV.02	Remote Verbindung	Mittel
SE-KExDaV.03	Parameter Ermittlung	Mittel
SE-KExDaV.04	Austausch Daten	Mittel
SE-KExDaV.05	Austausch Parameter	Mittel
SE-KExDaV.06	Datenaustausch-Plugin	Mittel
SE-KExDaV.07	Austausch dynamische Objekte	Mittel
SE-KExDaV.08	Austausch dynamische Mengen	Mittel

Tabelle 5-2: Kritikalität der SW-Komponenten/SW-Module/Prozesse/Datenbanken der SW-Einheit KExDaV

5.2.5 Sonstige Entwurfsentscheidungen

Es wurden keine sonstigen Entwurfsentscheidungen getroffen.

5.3 Schnittstellen

5.3.1 Externe Schnittstellen der SW-Einheit

Die SW-Einheit KExDaV verfügt über folgende externe Schnittstellen:

- KExDaV – Starter
(Aufrufschnittstelle der Applikation)
- Datenverteiler Applikationsfunktionen – Applikation
(Schnittstelle zur SW-Einheit „Datenverteiler-Applikationsfunktionen“ s. [TAnfDaV])
- KExDaV – Applikation
(logische Schnittstelle zu anderen Applikationen)

5.3.2 Interne Schnittstellen der SW-Einheit

Die SW-Einheit KExDaV verfügt über folgende interne Schnittstellen:

- Schnittstelle Verwaltung – Remote Verbindung
Beteiligte Elemente
 - Modul Verwaltung
 - Modul Remote Verbindung
- Schnittstelle Verwaltung – Parameter Ermittlung
Beteiligte Elemente
 - Modul Verwaltung
 - Modul Parameter Ermittlung

- Schnittstelle Remote Verbindung- Austausch Daten
Beteiligte Elemente
 - Modul Remote Verbindung
 - Modul Austausch Daten
- Schnittstelle Remote Verbindung- Austausch Parameter
Beteiligte Elemente
 - Modul Remote Verbindung
 - Modul Austausch Parameter
- Schnittstelle Remote Verbindung- Austausch dynamische Objekte
Beteiligte Elemente
 - Modul Remote Verbindung
 - Modul Austausch dynamische Objekte
- Schnittstelle Remote Verbindung- Austausch dynamische Mengen
Beteiligte Elemente
 - Modul Remote Verbindung
 - Modul Austausch dynamische Mengen
- Schnittstelle Austausch Daten/Parameter - Datenaustausch-Plugin
Beteiligte Elemente
 - Modul Austausch Daten/Parameter
 - Modul Datenaustausch-Plugin

5.4 Anforderungszuordnung

In Tabelle 5-3 ist die Zuordnung der Anforderungen aus Kapitel 4 "Technische Anforderungen an die SW-Einheit "KEx-DaV"" auf die SW-Einheit, Komponenten, Subkomponenten bzw. Module angegeben:

Anforderung	Modul SE-KExDaV.							
	01	02	03	04	05	06	07	08
TAnf-KExDaV 1: Allgemeine Technische Anforderungen	x	x				x		
TAnf-KExDaV 2: Start und Initialisierung	x					x		
TAnf-KExDaV 3: Verbindungsaufbau Remote DaV	x	x	x					
TAnf-KExDaV 4: Verbindungsabbruch Lokal-System	x							
TAnf-KExDaV 5: Austausch von Onlinedaten				x				
TAnf-KExDaV 6: Konfigurationsobjektauswahl			x					
TAnf-KExDaV 7: Korrespondierende Konfigurationsobjekte		x						
TAnf-KExDaV 8: Auswahl Onlinedaten			x					
TAnf-KExDaV 9: Unterschiedliche Konfigurationsstände						x		
TAnf-KExDaV 10: Referenzen in Onlinedatensätzen		x				x		
TAnf-KExDaV 11: Austausch von Parameterdaten					x			
TAnf-KExDaV 12: Konfigurationsobjektauswahl			x					
TAnf-KExDaV 13: Parameterdaten			x					
TAnf-KExDaV 14: Verwaltung Parameter Lokal-System (r)					x			
TAnf-KExDaV 15: Verwaltung Parameter Lokal-System (rw)					x			
TAnf-KExDaV 16: Verwaltung Parameter Remote-System (r)					x			
TAnf-KExDaV 17: Verwaltung Parameter Remote-System (rw)					x			
TAnf-KExDaV 18: Verwaltung Parameter Lokal- und Remote-System					x	x		
TAnf-KExDaV 19: Verwaltung Parameter Lokal- und Remote-System bei Trigger			x		x			
TAnf-KExDaV 20: TriggerKExDaV	x	x	x		x			
TAnf-KExDaV 21: Unterschiedliche Konfigurationsstände						x		
TAnf-KExDaV 22: Referenzen in Parameterdatensätzen		x				x		
TAnf-KExDaV 23: Plugin-Schnittstelle						x		
TAnf-KExDaV 24: Austausch von dynamischen Objekten			x				x	
TAnf-KExDaV 25: Initialisierung	x	x	x				x	
TAnf-KExDaV 26: Laufender Betrieb		x						
TAnf-KExDaV 27: Parameteränderungen			x				x	
TAnf-KExDaV 28: Kopieren dynamische Objekte			x					
TAnf-KExDaV 29: Löschen dynamische Objekte								
TAnf-KExDaV 30: Austausch von dynamischen Mengen								x
TAnf-KExDaV 31: Spezifikation Austausch			x					x
TAnf-KExDaV 32: Initialisierung		x						x
TAnf-KExDaV 33: Laufender Betrieb								x
TAnf-KExDaV 34: Parameteränderung			x					x
TAnf-KExDaV 35: Schnittstelle KExDaV – Starter	x							
TAnf-KExDaV 36: Schnittstelle KExDaV – Applikation	x	x	x					x

Tabelle 5-3: Zuordnung der Anforderungen an die SW-Module der SW-Einheit KExDaV

6 Prüfspezifikation SW-Einheit KExDaV

6.1 Anforderungen

6.1.1 Einstufung der Funktionseinheit hinsichtlich Kritikalität und IT-Sicherheit

Die Kritikalität dieser Funktionseinheit ist als **mittel** eingestuft.

6.1.2 Prüfanforderungen

Folgende Anforderungen müssen die Prüfungen erfüllen:

- Die Prüfungen müssen mit Normal-, Grenz- und fehlerhaften Werten durchgeführt werden.
- Die Prüfungen müssen möglichst mit allen Ausführungsoptionen durchgeführt werden.

6.2 Methoden der Prüfung

Folgende Methoden werden zur Durchführung der Prüfung eingesetzt:

- Statische Analyse (STAT)

Das Grundprinzip der statischen Analyse besteht darin, dass ein Prüfgegenstand, der nach einem vorgegebenen Formalismus aufgebaut ist, gelesen wird. Hierbei werden entweder sofort Fehler bzw. fehlerträchtige Situationen festgestellt bzw. Informationen abgeleitet, die nach Ende des Lesevorgangs Rückschlüsse auf Fehler bzw. fehlerträchtige Situationen zulassen.

- Black-Box-Testfallentwurf (BBTE)

Beim Black-Box-Testfallentwurf werden die Testfälle aus den Anforderungen bzw. Spezifikationen abgeleitet. Der Prüfgegenstand wird als schwarzer Kasten angesehen, d. h. der Prüfer ist nicht an der internen Struktur und dem Verhalten des Prüfgegenstandes interessiert. Dabei werden folgende Blackbox-Testfallentwurfsmethoden verwendet:

- Intuitive Testfallermittlung

Grundlage für diesen methodischen Ansatz ist die intuitive Fähigkeit und Erfahrung von Menschen, Testfälle nach erwarteten Fehlern auszuwählen.

- Funktionsabdeckung

Bei der Funktionsabdeckung werden Testfälle identifiziert, mit denen nachgewiesen werden kann, dass die jeweilige Funktion vorhanden und auch ausführbar ist. Hierbei wird der Testfall auf das Normalverhalten und das Ausnahmeverhalten des Prüfgegenstandes ausgerichtet.

Die Ergebnisse der Testfälle werden im Prüfprotokoll chronologisch nach Prüffällen gegliedert festgehalten.

- Review (REV)

Während der gesamten Testphase werden Reviews zur Identifikation von Fehlern, Feststellung von Verstößen gegen Spezifikationen, Standards und Pläne durchgeführt, um den Entwicklungsprozess zu verbessern und zu korrigieren.

6.3 Prüfkriterien

Prüfkriterium für den Black-Box-Test der SW-Einheit ist die vollständige Übereinstimmung der vom Prüfling im Rahmen des Testablaufs erzeugten Ausgaben mit den Sollvorgaben der Prüffallbeschreibungen.

6.3.1 Abdeckungsgrad

Die Anforderungen aus den Anwenderforderungen und technischen Anforderungen werden alle mindestens einmal geprüft. Dabei werden möglichst realitätsnahe komplexe Prüfscenarien untersucht, die das fehlerfreie Zusammenspiel der einzelnen Anforderungen sicherstellen.

6.3.2 Checklisten

Nachfolgend sind die grundlegenden Fragestellungen für die Checkliste zur Prüfung der SW-Segmente bzw. SW-Einheiten formuliert.

- Ist das Produkt nach dem Produktschema aufgebaut?
- Enthält das Produkt keine Syntaxfehler (z. B. Schreibfehler)?
- Sind in dem Produkt keine widersprüchlichen Aussagen?
- Sind alle Aussagen in dem Produkt eindeutig formuliert?
- Ist das Produkt vollständig?
- Sind in dem Produkt alle nach dem Produktschema relevanten Inhalte in adäquater Ausführlichkeit vorhanden?
- Ist das Produkt konsistent mit allen Vorgänger-Produkten, aus denen das zu prüfende Produkt hervorging?
- Ist das Produkt frei von Inkonsistenzen und Widersprüchen zu "Nachbar"-Produkten, die mit ihm in Beziehung stehen?
- Wird ein einheitlicher Modulrahmen verwendet?
- Wurden die Programmier-/Codierstandards eingehalten?
- Wurde der Code ausreichend und verständlich dokumentiert?

6.3.3 Endekriterien

Die Prüfung gilt als erfolgreich, wenn alle in Kapitel 6.4 aufgeführten Prüffälle mit den vorgegebenen Ergebnissen durchgeführt wurden.

6.4 Prüffälle

Die Prüffälle für die SW-Einheit KExDaV ergeben sich aus Kapitel 3 "Anwenderforderungen an die SW-Einheit "KEx-DaV"" und Kapitel 4 "Technische Anforderungen an die SW-Einheit "KEx-DaV".

Die Prüffälle, mit denen sichergestellt wird, dass die Anforderungen aus den oben aufgeführten Produkten erfüllt werden, sind in der folgenden Prüffallbeschreibung aufgeführt.

6.4.1 Prüffallbeschreibung

6.4.1.1 Prüffall 1: Review

Folgende Anforderungen sind (teilweise) durch Reviews zu prüfen:

Anforderung	Zu prüfender Aspekt
-------------	---------------------

TAnf-KExDaV 36	Die in den Technischen Anforderungen spezifizierten Attributgruppen sind so wie beschrieben umgesetzt:
----------------	--

- | |
|--|
| <ul style="list-style-type: none"> Abbildung 4-2: Parameterattributgruppe SpezifikationKExDaV Abbildung 4-11: Attributgruppe TriggerKExDaV (s.a. TAnf-KExDaV 20) |
|--|

6.4.1.2 Prüffall 2: TestKExDaV

Mit diesem Prüffall wird die grundlegende Funktionalität von KExDaV getestet,

Dazu werden die folgenden Prüfungen durchgeführt.

2.1 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Festlegung eines änderbaren Konfigurationsbereiches
- Start von KExDaV
- Erstellung dynamischer Objekte im Konfigurationsbereich (KB).
Der Konfigurationsbereich hat 6 dynamische Objekte, von denen 2 eine PID aufweisen. Die restlichen 4 Objekte haben keine PID. Zwei von ihnen wurden kopiert bzw. ausgetauscht.
- Definition Parameter unter Angabe des KB als Standardaustauschbereich

Prüfungen:	Erwartete Ergebnisse
Durch die Setzung des Parameters muss der Austauschbereich aktualisiert werden. Daher müssen die auf der Quellseite nicht vorhandenen Objekte gelöscht werden.	Der Konfigurationsbereich enthält nur noch die beiden nicht ausgetauschten dynamischen Objekte.

2.2 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Definition Parameter für einen einfachen Austausch von Onlinedaten
- Sendung von Online-Daten
- Änderung des Parameters
- Sendung von Online-Daten

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die Onlinedaten auf dem anderen Datenverteiler ankommen. Prüfung, ob die Parameteränderung wirksam ist	Daten werden übertragen, solange der Parameter so gesetzt ist.

2.3 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Definition Parameter für einen einfachen Austausch von Onlinedaten mit Referenzen
- Sendung von Online-Daten
- Änderung des Parameters
- Sendung von Online-Daten

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die Onlinedaten auf dem anderen Datenverteiler ankommen.	Daten werden übertragen, solange der Parameter so gesetzt ist.
Prüfung, ob die Parameteränderung wirksam ist.	Referenzwerte werden korrekt übertragen
Prüfung, ob die Referenzwerte entsprechend der Anforderungen übertragen wurden (dynamische Objekte werden kopiert, sofern eine Pid vorhanden ist, optionale Referenzattribute werden leer gelassen wenn kein korrespondierendes Objekt gefunden wurde, bei nicht optionalen Referenzattributen wird bei fehlenden Objekten ein leerer Datensatz gesendet.	

2.4 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Definition Parameter für einen einfachen nur lesenden Austausch von Parameterdaten
- Sendung Parameter-Daten auf dem System das die Parameter verwaltet
- Sendung Parameter-Daten auf dem System das nur Lesen darf

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die Parameter auf dem anderen Datenverteiler ankommen.	Parameterdaten in die korrekte Richtung werden übertragen, in Gegenrichtung nicht.
Prüfung, ob die Parameter in Gegenrichtung auf dem ersten Datenverteiler ankommen	

2.5 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Definiert Parameter für einen lesenden und schreibenden Austausch von Parameterdaten
- Sendung Parameter-Daten auf dem System das die Parameter verwaltet
- Sendung Parameter-Daten auf dem System das die Parameter nicht verwaltet

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die Parameter auf dem anderen Datenverteiler ankommen. Prüfung, ob die Parameter in Gegenrichtung auf dem ersten Datenverteiler ankommen	Parameterdaten können von beiden Systemen geschrieben und gelesen werden.

2.6 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Startet Sender und Empfänger für die Simulation der Parametrierung.
- Start von KExDaV
- Definition Parameter für einen Parameteraustausch mit beidseitiger Verwaltung
- Sendung Parameter-Daten auf dem System 1
- Sendung Parameter-Daten auf dem 2

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die Parameter jeweils auf dem anderen Datenverteiler unter dem Vorgabe-Aspekt ankommen. Prüfung, dass die Parameterdaten in der Übertragung so ausgebremst werden, dass kein ständiges hin und herschalten wie in TAnf-KExDav 18 beschrieben passieren kann.	Parameterdaten können von beiden Systemen geschrieben und gelesen werden. Das gleichzeitige Senden von beiden Systemen wird unterdrückt

2.7 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Startet Sender und Empfänger für die Simulation der Parametrierung.
- Start von KExDaV
- Definition Parameter für einen Parameterraustausch mit beidseitiger Verwaltung mit Trigger
- Sendung Parameter-Daten auf dem System 1
- Triggerung
- Sendung Parameter-Daten auf dem 2
- Triggerung

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die Parameter jeweils auf dem anderen Datenverteiler unter dem Vorgabe-Aspekt ankommen, aber nur wenn vorher die Triggerung erfolgte. Prüfung, dass nach einer Triggerung die Parameter nur einmal übertragen werden.	Parameterdaten können von beiden Systemen geschrieben und gelesen werden. Die Parameterdaten werden nur übertragen nachdem vorher die Triggerung erfolgte Durch die Triggerung werden die Parameterdaten nur einmal ausgetauscht.

2.8 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Definition Verbindungsparameter
- Stopp von Datenverteiler 2
- Warten
- Neustart von Datenverteiler 2

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob erkannt wird, wenn die Verbindung zu einem Datenverteiler abbricht und das KExDaV die Verbindung wieder aufbaut, wenn der Datenverteiler wieder erreichbar ist	KExDaV merkt wenn die Verbindung zum Datenverteiler unterbrochen ist und verbindet sich erneut wenn sie wieder verfügbar ist.

2.9 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Erzeugung eines dynamischen Objektes 1
- Definition Parameter für den Austausch von Objekten
- Erzeugung eines dynamischen Objektes 2
- Konfigurationsdaten für das dynamische Objekte 2 erstellen
- Löschen (Ungültig machen) des dynamischen Objektes 1
- Parameter ändern. Den weiteren Austausch deaktivieren.
- Erzeugung eines dynamischen Objektes 3

Prüfungen:	Erwartete Ergebnisse
Prüfung, ob die dynamischen Objekte 1 und 2 sowie die Konfigurationsdaten korrekt übertragen werden.	Die dynamischen Objekte 1 und 2 wurden korrekt übertragen.
Prüfung, ob das Löschen des dynamischen Objekte 1 bewirkt, dass dieses auch auf dem anderen Datenverteiler gelöscht wird.	Das dynamische Objekt 1 wird gelöscht.
Prüfung, ob die Parameteränderungen sich auswirkt. Das dynamische Objekt 3 darf nicht übertragen werden.	Das dynamische Objekt 3 wird nicht übertragen.

2.10 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Erstellung einer Menge mit einem dynamischen Objekt auf Datenverteiler 1 und eine Menge mit einem anderen dynamischen Objekt auf Datenverteiler 2
- Der Parameter wird so eingestellt, dass die Menge von Datenverteiler 1 auf Datenverteiler 2 zu übertragen ist.
- Der Menge wird ein Objekt hinzugefügt und danach wird ein Objekt aus der Menge entfernt.
- Der Parameter wird erneut geändert. Dabei wird die weitere Übertragung unterbunden.
- Der Menge wird ein weiteres Objekt hinzugefügt.

Prüfungen:	Erwartete Ergebnisse
Prüfung, dass nur Objekte von Datenverteiler 1 nach Datenverteiler 2 übertragen werden.	Änderungen an der lokalen Menge werden entsprechend übertragen.
Prüfung, dass sich die Menge von Datenverteiler 2 entsprechend der Änderungen ändert (Solange der Parameter dies vorgibt).	Nach der Änderung des Parameters wird die Übertragung gestoppt.

2.11 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Erstellung einer Menge mit zwei dynamischen Objekten auf Datenverteiler 1 und einer Menge mit einem dynamischen Objekt auf Datenverteiler 2, welches zu einem Objekt von Datenverteiler 1 identisch ist.
- Der Parameter wird so eingestellt, dass die Menge von Datenverteiler 1 auf Datenverteiler 2 zu übertragen ist.

Prüfungen:	Erwartete Ergebnisse
Prüfung, dass nur Objekte von Datenverteiler 1 nach Datenverteiler 2 übertragen werden.	Änderungen an der lokalen Menge werden entsprechend übertragen.
Prüfung, dass sich die Menge von Datenverteiler 2 entsprechend der Änderungen ändert.	Objekte, die bereits in der Zielmenge existieren werden nicht erneut kopiert oder zwischenzeitlich gelöscht.

2.12 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Erstellung einer Menge mit einem dynamischen Objekt und einem Konfigurationsobjekt auf Datenverteiler 1 und einer Menge mit einem dynamischen Objekt auf Datenverteiler 2, welches zu dem Objekt von Datenverteiler 1 identisch ist.
- Der Parameter wird so eingestellt, dass die Menge von Datenverteiler 1 auf Datenverteiler 2 zu übertragen ist.

Prüfungen:	Erwartete Ergebnisse
Prüfung, dass nur Objekte von Datenverteiler 1 nach Datenverteiler 2 übertragen werden.	Objekte, die bereits in der Zielmenge existieren werden nicht erneut kopiert oder zwischenzeitlich gelöscht.
Prüfung, dass sich die Menge von Datenverteiler 2 entsprechend der Änderungen ändert.	Das Konfigurationsobjekt wird nicht übertragen.

2.13 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Erstellung einer Menge mit 5 dynamischen Objekten auf Datenverteiler 1 und einer Menge auf Datenverteiler 2, die auf 3 Objekte beschränkt ist.
- Der Parameter wird so eingestellt, dass die Menge von Datenverteiler 1 auf Datenverteiler 2 zu übertragen ist.

Prüfungen:	Erwartete Ergebnisse
Prüfung, dass nur Objekte von Datenverteiler 1 nach Datenverteiler 2 übertragen werden. Prüfung, dass sich die Menge von Datenverteiler 2 entsprechend der Änderungen solange wie möglich ändert.	Es werden nur die ersten drei Objekte übertragen, da die Zielmenge die Beschränkung aufweist. KExDaV darf nicht abstürzen.

2.14 Ausgangslage / Ablauf

- Start einer Umgebung mit 2 Datenverteilern
- Start von KExDaV
- Erstellung einer Menge mit einem dynamischen Objekt ohne PID auf Datenverteiler 1 und einer Menge auf Datenverteiler 2, mit einem anderen dynamischen Objekt.
- Der Parameter wird so eingestellt, dass die Menge von Datenverteiler 1 auf Datenverteiler 2 zu übertragen ist.
- Ein weiteres Objekt ohne PID wird der Menge auf Datenverteiler 1 zugefügt und anschließend wird ein Objekt der Menge wieder entfernt.

Prüfungen:	Erwartete Ergebnisse
Prüfung, dass nur Objekte von Datenverteiler 1 nach Datenverteiler 2 übertragen werden.	Es werden alle Objekte (mit und ohne PID) übertragen.

6.4.1.3 Prüffall 3: TestKExDaVObjekt

Mit diesem Prüffall werden alle Methoden getestet, die die interne Verwaltung eines KExDaV-Objektes betreffen.

Prüfungen:

- Anmeldungen von Daten
- Senden und Empfangen von Daten
- Prüfung auf Existenz und Benachrichtigungen
- Methoden zum Erstellen und Löschen des Objekts
- Konfigurationsdaten ermitteln
- Typ, Pid, Verbindung und internes Systemobjekt ermitteln

Erwartete Ergebnisse:

Die einzelnen Methoden erfüllen die gestellten Anforderungen.

6.4.1.4 Prüffall 4: TestAdjustableTimer

Mit diesem Prüffall wird die Wartezeit zwischen zwei Verbindungsversuchen getestet.

6.4.1.5 Prüffall 5: TestBasicKExDaVDataPlugin

Mit diesem Prüffall wird die Standardimplementierung der PlugIn-Schnittstelle zum Kopieren von Datensätzen getestet.

Prüfungen:	Erwartete Ergebnisse
Kopieren von Daten in identischen Attributgruppen.	Alle Daten werden erfolgreich kopiert.
Kopieren von Daten in eine Attributgruppe, die nicht alle Attribute der Quellattributgruppe enthält.	Alle vorhandenen Daten werden übertragen. Die, in der Zielattributgruppe nicht vorhandenen, Attribute werden ignoriert.
Kopieren von Daten in eine Attributgruppe, die mehr Attribute als die Quellattributgruppe enthält.	Alle Attribute, zu denen keine Daten im Quelldatensatz vorhanden sind, werden auf ihren Standardwert gesetzt (falls möglich).
Kopieren von Attributen, die Referenzen auf Objekte enthalten.	Die Referenzen werden kopiert und ggf. werden auf dem Zielsystem neue Objekte erstellt.

6.4.1.6 Prüffall 6: TestLowLevelDataPipe

Prüft die einfache Übertragung von Daten zwischen 2 Systemen

- mit gleichem Objekt
- mit unterschiedlichem Objekt
- mit unterschiedlicher Simulationsvarianten

Erwartete Ergebnisse:

Die Übertragungen werden ordnungsgemäß durchgeführt.

6.4.1.7 Prüffall 7: TestKExDaVLocalApplication

Prüffälle zur ordnungsmäßigen Verarbeitung der Aufrufparameter.

6.4.1.8 Prüffall 8: TestKExDaVWrappedReferenceValue

Prüffall, dass bei dem Kopieren von Data-Objekten das Setzen von Objekten nur über die Pid möglich ist, und somit eine zwingende Trennung zwischen den Datenverteilern für den Datenaustausch besteht.

6.4.1.9 Prüffall 9: TestApplicationStartExit

Prüffall, dass sich KExDaV beim Terminieren der lokalen Verbindung beendet.

6.4.2 Abdeckungsmatrix

Tabelle 6-1 dokumentiert die Abdeckung der fachlichen und technischen Anforderungen an den Prüfgegenstand durch die einzelnen Prüffälle.

Anforderung	1	2	3	4	5	6	7	8	9
Afo-1		x							
Afo-2		x							
Afo-3		x							
TAnf-KExDaV 1: Allgemeine Technische Anforderungen		x						x	
TAnf-KExDaV 2: Start und Initialisierung		x		x					
TAnf-KExDaV 3: Verbindungsaufbau Remote DaV		x							
TAnf-KExDaV 4: Verbindungsabbruch Lokal-System									x
TAnf-KExDaV 5: Austausch von Onlinedaten		x				x			
TAnf-KExDaV 6: Konfigurationsobjektauswahl									
TAnf-KExDaV 7: Korrespondierende Konfigurationsobjekte			x						
TAnf-KExDaV 8: Auswahl Onlinedaten		x				x			
TAnf-KExDaV 9: Unterschiedliche Konfigurationsstände					x				
TAnf-KExDaV 10: Referenzen in Onlinedatensätzen		x			x				
TAnf-KExDaV 11: Austausch von Parameterdaten		x				x			
TAnf-KExDaV 12: Konfigurationsobjektauswahl									
TAnf-KExDaV 13: Parameterdaten		x				x			
TAnf-KExDaV 14: Verwaltung Parameter Lokal-System (r)		x				x			
TAnf-KExDaV 15: Verwaltung Parameter Lokal-System (rw)		x				x			
TAnf-KExDaV 16: Verwaltung Parameter Remote-System (r)		x				x			
TAnf-KExDaV 17: Verwaltung Parameter Remote-System (rw)		x				x			
TAnf-KExDaV 18: Verwaltung Parameter Lokal- und Remote-System		x				x			
TAnf-KExDaV 19: Verwaltung Parameter Lokal- und Remote-System bei Trigger		x				x			
TAnf-KExDaV 20: TriggerKExDaV	x	x							
TAnf-KExDaV 21: Unterschiedliche Konfigurationsstände		x						x	
TAnf-KExDaV 22: Referenzen in Parameterdatensätzen					x				
TAnf-KExDaV 23: Plugin-Schnittstelle		x			x				
TAnf-KExDaV 24: Austausch von dynamischen Objekten		x							
TAnf-KExDaV 25: Initialisierung		x							
TAnf-KExDaV 26: Laufender Betrieb		x							
TAnf-KExDaV 27: Parameteränderungen		x							
TAnf-KExDaV 28: Kopieren dynamische Objekte		x							
TAnf-KExDaV 29: Löschen dynamische Objekte			x						
TAnf-KExDaV 30: Austausch von dynamischen Mengen		x							
TAnf-KExDaV 31: Spezifikation Austausch		x							
TAnf-KExDaV 32: Initialisierung		x							
TAnf-KExDaV 33: Laufender Betrieb		x							
TAnf-KExDaV 34: Parameteränderung		x							
TAnf-KExDaV 35: Schnittstelle KExDaV – Starter	x						x		
TAnf-KExDaV 36: Schnittstelle KExDaV – Applikation	x								

Tabelle 6-1: Abdeckungsmatrix