VDV Die Verkehrsunternehmen

VDV-Schrift 301-2

07/2018

IBIS-IP Beschreibung der Dienste / Service description

Basisdienste / Base Services
DeviceManagementService, SystemManagementService, SystemDocumentationService
V2.1

Gesamtbearbeitung
Ausschuss für Telematik und Informationssysteme (ATI)

Gefördert durch:

Bundesministerium für Wirtschaft und Energie

aufgrund eines Beschlusses des Deutschen Bundestages

Das dieser VDV-Schrift zugrundeliegende Vorhaben IP-KOM-ÖV wurde mit Mitteln des Bundesministeriums für Wirtschaft und Energie unter dem Förderkennzeichen 19P10003 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

IBIS-IP Beschreibung der Dienste / Service description

Basisdienste / Base Services
DeviceManagementService, SystemManagementService, SystemDocumentationService
V2.1

Sachbearbeitung
Unterausschuss für Telematik
(UA Telematik)

Autorenverzeichnis
Dipl.-Ing. Dirk Weißer, INIT, Karlsruhe
Dr. Torsten Franke, IVU, Aachen
Dr. Holger Bandelin, Scheidt & Bachmann, Mönchengladbach
Dipl.-Ing. Berthold Radermacher, VDV, Köln
Dipl.-Ing. (FH) Andreas Wehrmann, VDV, Köln
Dipl.-Ing. ETH Walter Meier-Leu, we, Schaffhausen
Dipl.-Ing. René Fischli, Trapeze, Neuhausen
Dipl.-Ing. Peter Schüssler, DResearch FE, Berlin
Dr. Bernd Schubert, iris-GmbH, Berlin

Vorwort

Auf Initiative des VDV und gefördert durch das BMWi begann im September 2010 das Forschungs- und Standardisierungsprojekt *Internet Protokoll basierte Kommunikationsdienste im öffentlichen Verkehr (IP-KOM-ÖV)*.

Das Projekt wird von 14 Partnern aus Industrie, Universitäten und Verkehrsunternehmen getragen. Es dient der Erarbeitung moderner Kommunikationskonzepte für die umfassende und kontinuierliche Fahrgastinformation.

Eine umfassende Fahrgastinformation stellt heutzutage ein entscheidendes Wettbewerbsmerkmal im öffentlichen Personenverkehr dar, nicht nur im Vergleich mit anderen Verkehrsunternehmen sondern auch im Vergleich zum Individualverkehr.

Bereits heute ist es üblich, dass Verkehrsunternehmen ihre Fahrgäste nicht nur über die geplanten Fahrten informieren, sondern auch Echtzeitinformationen z. B. zu Verspätungen, Ereignisse oder Fahrtzieländerungen bereitstellen. Diese Informationen werden zum einen über öffentliche Anzeiger bzw. Ansagen in Fahrzeugen oder an Haltestellen allen dort befindlichen Personen zur Verfügung gestellt. Zum anderen lassen sich solche Informationen mit speziellen Applikationen oder über Web-Angebote von Einzelpersonen individuell abfragen.

Bislang ist es aber nicht möglich, Fahrgäste im öffentlichen Verkehr mit Informationen zu ihrer persönlich relevanten Fahrt zu versorgen, den Fahrgast also auch im Störungsfall mit Hilfe des öffentlichen Verkehrs auf dem schnellsten Weg zu seinem Ziel zu führen.

Die weit verbreiteten Smartphones und Tablets bieten hierfür vielfältige Möglichkeiten und ermöglichen eine hohe Akzeptanz der Benutzer. Die Informationsübertragung erfolgt dabei IP-basiert und sollte bevorzugt zwischen einem zentralen Informations-Server und dem Kundenendgerät erfolgen. Für den Fall, dass der zentrale Datenserver nicht erreichbar ist, resp. das Fahrzeug nicht mit einem solchen verbunden ist, sollte auch eine Kommunikation direkt zwischen Kundenendgerät und Fahrzeug möglich sein.

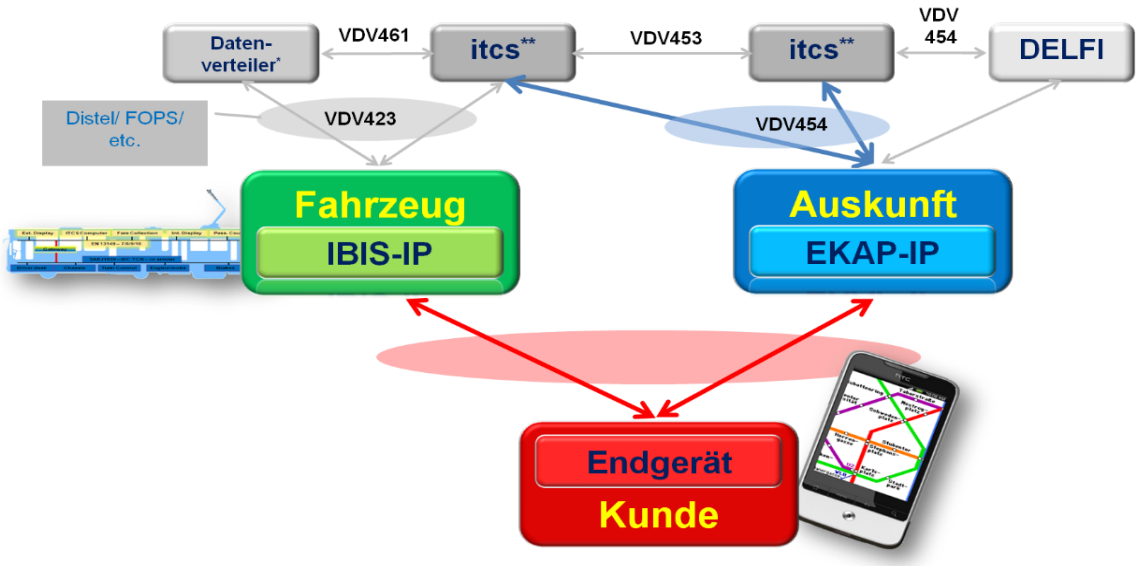

Abbildung 1: Umfeld und Schwerpunkte im Projekt IP-KOM-ÖV

Das Forschungs- und Standardisierungsprojekt IP-KOM-ÖV arbeitet deshalb an drei Schwerpunkten (vgl. Abbildung 1).

Erster Schwerpunkt (grün in Abbildung 1) ist die Spezifikation eines performanten IP-basierten Kommunikationsprotokolls im Fahrzeug (IBIS-IP). Dabei geht es zum einen darum, den gewachsenen Bedürfnissen der fahrzeuginternen Fahrgastinformation und Kommunikation gerecht zu werden. Hierzu wird der in den achtziger Jahren entwickelte IBIS-Wagenbus aus der VDV-Schrift 300 auf eine moderne IP-Informationsarchitektur umgesetzt. Zum anderen geht es um die Definition einer IP-basierten Schnittstelle zur Übertragung der Informationen vom Fahrzeug zum mobilen Kundenendgerät. Die vorliegende Schrift ist das Resultat dieses Teils des Forschungsprojektes.

Zweiter Schwerpunkt (rot in Abbildung 1) ist die Definition der notwendigen Schnittstellen um zukünftigen Applikationen für individuelle Fahrgastinformation unter Verwendung mobiler Geräte des Fahrgasts (Smartphones, Tablet-PC u. ä.) standardisiert zur Verfügung zu stellen. Hierzu wurden im ersten Schritt die Bedürfnisse von Fahrgästen zu individuellen Informationen ermittelt. Im zweiten Schritt wurden einheitliche Schnittstellen zwischen der Echtzeit-Kommunikations- und Auskunftsplattform (EKAP vgl. dritten Schwerpunkt) und den mobilen Kundenendgeräten bzw. zwischen der EKAP und den Hintergrundsystemen entwickelt. Hierbei wurden ausschließlich die Datenmodellierungen und Architekturen erforscht und spezifiziert. Die Entwicklung einer Applikation für mobile Endgeräte war ausdrücklich nicht vorgesehen und wurde lediglich für Tests in einfachster Form realisiert. Die Resultate dieses Teils des Forschungsprojektes sind in der VDV-Schrift 430 dokumentiert.

Dritter Schwerpunkt (blau in Abbildung 1) ist die Definition und Schaffung einer EKAP Die EKAP bündelt Informationen von itcs- und anderen Auskunfts- und Informationssystemen und stellt die Vielzahl an Informationen über geeignete Schnittstellen den Applikationen auf den Kundenendgeräten standardisiert zur Verfügung. Diese Plattform ermöglicht es den Herstellern von Apps, Kunden dynamisch mit individuellen Störungsmeldungen versorgen zu können. Die Resultate dieses Teils des Forschungsprojektes sind ebenfalls in der VDV-Schrift 431 dokumentiert.

Darüber hinaus wird die Praxistauglichkeit dieses neuen Standards in Labor- und Feldtests verifiziert.

Im Zuge der Weiterentwicklung der VDV-301 hat sich gezeigt, dass die Integration aller Dienste in einer einzigen Schrift 301-2 Version 1.0 umständlich in der Handhabung ist. Daher wurden die einzelnen Dienste in einzelne Schriften separiert.

Die Unterteilung der Dienste in einzelne Dokumente ist begründet in der einfacheren Anpassung und Freigabe. Zurzeit sind viele Hersteller daran, die Norm in die Umsetzung zu bringen. Daher ist ein grösserer Anpassungsbedarf vorhanden. Auch können neue Dienste unabhängig von den bestehenden angepasst werden.

Die VDV-301-Schriften bestehen aus grundlegenden Dokumenten wie die VDV-301-1 und VDV-301-2 sowie die einzelnen Beschreibungen der Dienste (VDV-301-2-x).

In der VDV-301-2 werden die technischen Grundlagen wie auch die Basisdienste, welche die Grundlagen eines IBIS-IP-Systems bilden, beschrieben. Unter den Basisdiensten werden die Dienste DeviceManagementService, SystemManagementService und SystemDocumentationService verstanden.

Inhaltsverzeichnis

Abkürzungen

Die bereits in der VDV 301-1 definierten Abkürzungen werden an dieser Stelle nicht wiederholt.

The abbreviations already defined in VDV 301-1 are not repeated here.

Abkürzung / Abreviation	**Beschreibung**	**Description**
APC	Fahrgastzählsystem Beschreibt die Geräteklasse des Fahrgastzählsystems innerhalb des Fahrzeuges. Die technische Umsetzung der Zählung wird nicht unterschieden.	Automatic passenger counting system Describes the device class of the passenger counting system within the vehicle. The technical implementation of the count is not differentiated.
DNS	Domain Name Server	Domain Name Server
DNS-SD	Domain Name Server Service Discovery	Domain Name Server Service Discovery
EEPROM	Electrically Erasable Programmable Read-Only Memory	Electrically Erasable Programmable Read-Only Memory
EKAP	Echtzeit Kommunikations- und Auskunftsplattform. Siehe VDV 431-1.	Real-Time Communication and Information Platform EKAP cf. VDV 431-1.
GNSS	Global Navigation Satellite System	Global Navigation Satellite System
GPS	Global Positioning System	Global Positioning System
IETF	Internet Engineering Task Force	Internet Engineering Task Force
RFC	„Reference“ Technisches Datenblatt der Internet Society	„Reference“ technical datasheet oft he Internet Society
SNTP	Simple Network Time Protocol	Simple Network Time Protocol
XSD	XML Schema Definition	XML Schema Definition
DELFI	Deutschlandweite Elektronische Fahrplaninformation	Germanwide Journey Information system
UML	Unified Modeling Language	Unified Modeling Language

1 Einführung in IBIS-IP

Zu Beginn dieser Schrift werden einige einführende Definitionen zur Umsetzung beschrieben. Die Kenntnis des Teil 1 (VDV- Schrift 301-1) wird hierbei vorausgesetzt.

❖ **Introduction to IBIS-IP**

Several introductory implementation definitions are described at the beginning of this document. Knowledge of part 1 (VDV recommendation 301-1) is assumed.

1.1 Anforderungen an Geräte in IBIS-IP

In den nachfolgenden Abschnitten werden die Anforderungen an Geräte und Dienste beschrieben, die sich aus dem im Teil 1 (VDV- Schrift 301-1) skizzierten System ergeben.

Demnach muss ein Gerät folgende Mindestanforderungen erfüllen, um am IBIS-IP teilnehmen zu können:

— eine Ethernet-Schnittstelle

— Erkennung der Einbauposition

— eine TCP/IP- bzw. UDP/IP-Stack Implementierung

— Fähigkeit zur Verarbeitung von HTTP- Protokollen

— Speicher für die gerätespezifische Konfigurationsdaten

— Implementierung der DNS-SD Funktionalität

— Rückwirkungsfreiheit zu angeschlossenen System gewährleisten (falls Verbindung vorhanden)

— Bereitstellen eines `DeviceManagementService` (vgl. Kapitel 3.2)

— Optional: Bereitstellen einer webbasierten herstellerspezifischen Wartungsschnittstelle, die per URL (im `DeviceManagementService`) angesprochen werden kann.

❖ **Requirements on Devices in IBIS-IP**

The requirements on devices and services resulting from the system drafted in part 1 (VDV recommendation 301-1) are described in the following sections.

Thus, a device must meet the following minimum requirements to participate in IBIS-IP:

- Ethernet interface
- Detection of the installation position
- TCP/IP and/or UDP/IP stack implementation
- Ability to process HTTP protocols
- Memory for device-specific configuration data
- Implementation of DNS-SD functionality
- Ensures no feedback into the connected system (if connection exists)
- Provision of a `DeviceManagementService` (see chapter 3.2)

- *Optional: Provision of a web-based manufacturer-specific maintenance interface that can be addressed by URL (in `DeviceManagementService`).*

1.2 Konfigurationsparameter

In einem IBIS-IP-System ist ein großer Teil des Systemverhaltens konfigurierbar. So lässt sich durch die Systemkonfiguration festlegen, welche Dienste auf welchen Geräten in welcher IBIS-IP-Version laufen.

Eine IBIS-IP-Systemkonfiguration enthält also Angaben dazu,

— welche Geräte zum System gehören (bestehend aus Geräteklasse (vgl. Kapitel 1.4) und Einbaukennung (vgl. Kapitel 1.3),

— welche Dienste in welcher IBIS-IP-Version zum System gehören und

— auf welchem Gerät welcher Dienst läuft.

Fahrzeugspezifische Angaben (Fahrzeugnummer, Funkadresse und gerätespezifische Konfigurationsparameter) spielen dagegen für das IBIS-IP-System keine Rolle.

❖ Configuration Parameters

A large part of the system behavior can be configured in an IBIS-IP system. Using the system configuration, it can be defined, which services run on which devices in which IBIS-IP version.

Thus, an IBIS-IP system configuration contains information,

- which devices belong to the system (consisting of device class (cf. Chapter 1.4) and installation identifier (cf. chapter 1.3),
- which services belong to the system in which IBIS-IP version, and
- on which device the service is running.

Vehicle-specific information (vehicle number, radio address, and device-specific configuration parameters) are of no importance for the IBIS-IP system.

1.3 Einbaukennung

Die Ermittlung der Einbaukennung kann durch Auslesen einer Information erfolgen, die am Ort des Geräte-Einbaus hinterlegt ist. Auf welchem technischen Weg dies erfolgt (Steckercodierung, Eeprom, USB-Stick o.ä.), spielt dabei keine Rolle. Der `DeviceManagementService` eines Gerätes kann dann die Information über die Einbaukennung anderen Diensten bekanntmachen und ermöglicht so eine Zuordnung einer technischen Gerätekennung (IP-Adresse, DNS-Name) zu einer fachlichen Gerätekennung (aus Geräteklasse und Einbauposition).

❖ Installation Identifier

The installation identifier is used to determin the mounting positon of a device by reading information stored at the device installation location. The technical implementation (plug coding, EEPRROM, USB stick, or similar) is not important. The DeviceManagement-Service of a device can then communicate the information via the installation identifier to other services, and allows an

assignment of a technical device identifier (IP address, DNS name) to a functional device identifier (consisting of device class and installation position).

1.4 Geräteklassen

Zur eindeutigen Identifikation der angeschlossenen Geräte im IBIS-IP-System wurden Geräteklassen spezifiziert, die alle derzeit denkbaren Geräte abdecken. Dabei wurde die Klassifizierung bereits in englischer Sprache vorgenommen, da diese Begriffe als „enumeration“ im englisch gehaltenen XML (vgl. Kapitel 7ff) verbreitet werden.

Die in IBIS-IP verwendeten Geräteklassen sind wie folgt definiert:

Geräteklasse	**Gerät**	**Beschreibung**
OnBordUnit	Bordrechner	Entspricht dem Zentralgerät gemäß VDV 300.
Validator	Entwerter	Stempelautomat oder elektronischer Entwerter für E-Tickets.
SideDisplay	Seiten Anzeiger	Beschreibt die seitliche Aussenanzeige an einem Fahrzeug.
FrontDisplay	Front Anzeiger	Entspricht der Fahrzeugzielanzeige.
InteriorDisplay	Innen Displays	Beschreibt die Anzeiger im Innenraum des Fahrzeuges, dies können Fahrgastinformationsanzeiger aber auch digitale Werbeanzeiger sein.
TicketVendingMachine	Ticket Automat	Beschreibt Fahrscheinverkaufsgeräte innerhalb des Fahrzeuges. Dies können fahrerbediente aber auch durch Fahrgäste bediente Automaten sein.
AnnouncementSystem	ELA	Beschreibt die Elektroakustische-Anlage in Fahrzeugen
MMI	Fahrer (Bedien-) Display	Entspricht der Fahrerbedieneinheit mit Anzeigemöglichkeit (z. B. Touch-Display).
VideoSystem	Videoüberwachungssystem	Beschreibt das Videoüberwachungssystem, das über IBIS-IP gesteuert wird bzw. Videodaten aufzeichnet, anzeigt und/oder überträgt.
APC	Fahrgastzählsystem	Beschreibt das Fahrgastzähsystem innerhalb des Fahrzeuges. Die technische Umsetzung der Zählung wird nicht unterschieden.

Geräteklasse	Gerät	Beschreibung
MobileInterface	Kundenkommunikation s- Schnittstelle	Beschreibt die Schnittstelle, an der die Fahrgäste mit ihrem mobilen Endgerät Informationen vom Fahrzeug abholen können.
TestDevice	Test Geräte	Platzhalter für Testgeräte, die im Falle einer Verifikation des Systems angeschlossen werden.
Other	Sonstige Geräte	Hierunter lassen sich Geräte anbinden, die noch nicht in diesem Standard klassifiziert wurden.

Tabelle 1 IBIS-IP Geräteklassen Klassifikation (Sortierung beliebig)

Die Geräteklasse sorgt zusammen mit der Einbaukennung für eine eindeutige fachliche Identifikation eines Geräts in einem IBIS-IP-System (vgl. Kapitel 5.1.4.1)

❖ Device Classes

Device classes covering all currently conceivable devices were specified in the IBIS-IP system for the unique identification of the connected devices. The classification was already defined in English, as these terms are common as "enumeration" in the English-language XML (cf. chapter7pp).

The device classes used on IBIS-IP are defined as follows:

Device class	Device	Description
OnBoardUnit	On-board computer	Corresponds to the central unit according to VDV 300.
Validator	Ticket Validator	Stamping machine or electronic validator for E-tickets
SideDisplay	Side display	Describes the exterior side display on a vehicle.
FrontDisplay	Front display	Corresponds to the vehicle destination display.
InteriorDisplay	Interior displays	Describes the displays in the vehicle interior, these can be passenger information displays or digital advertisement displays.
TicketVendingMachine	Ticket vending machine	Describes the ticket vending machines inside the vehicle. These can be driver-operated as well as passenger-operated machines.
AnnouncementSystem	Electroacoustic system	Describes the electroacoustic system in vehicles
MMI	Driver (operation) display	Corresponds to the driver’s operating unit with display possibility (e.g. touch display).

Device class	Device	Description
VideoSystem	Video monitoring system	Describes the video monitoring system, which is controlled via IBIS-IP and/or transfers video data.
APC	Automated passenger counting system	Describes the passenger counting system inside the vehicle. The technical counting implementation is not differentiated.
MobileInterface	Customer communication interface	Describes the interface, where passengers can retrieve information from the vehicle using their mobile end devices.
TestDevice	Test devices	Placeholder for test devices, which will be connected for system verification.
Other	Other devices	Devices which were not classified yet in this standard can be connected with this class.

Table 2: IBIS-IP device class classification (any sorting)

Together with the installation identifier, the device class ensures the unique functional identification of a device in an IBIS-IP system (cf. Chapter 5.1.4.1.1)

1.5 Notationen

Im Sinne einer besseren Lesbarkeit werden im weiteren Verlauf des Dokuments folgende Schriftarten mit spezieller Bedeutung verwendet.

Operationen:

Für Operationen wird folgendes Format verwendet:

`Operation in Courier New, Schriftgrad 11 und Fett`

Dienste:

Für Dienste wird folgendes Format verwendet:

`Dienst in Courier New, Schriftgrad 11 und Kursiv`

Syntax:

Eine Syntax wird mit folgendem Format dargestellt:

`<Syntax in Courier New, Schriftgrad 11 und mit <> eingefasst>`

Pfad:

Ein Pfad wird im folgenden Format dargestellt:

`Pfad in Courier New, Schriftgrad 11 ohne vorausgehende und folgende /`

❖ Notations

For improved readability, the following fonts are used with special meaning in the further course of this document.

Operations:

The following format is used for operations:

`Operation in Courier New, font size 11 and bold`

Services:

The following format is used for services:

`Service in Courier New, font size 11 and italic`

Syntax:

A syntax is represented in the following format:

`<Syntax in Courier New, font size 11, and enclosed in <>>`

Path:

A path is represented in the following format:

`Path in Courier New, font size 11 without leading or trailing /`

1.6 Versionierung in IBIS-IP

Jeder Service von IBIS-IP wird unabhängig versioniert. Eine gemeinsame IBIS-IP Version gibt es daher nicht. In einem IBIS-IP System muss sichergestellt werden, dass Client und Server der verwendeten Dienste zu einander kompatibel sind.

❖ IBIS-IP- Version

Each service of IBIS-IP is versioned independently. There is therefore no common IBIS-IP version. In an IBIS-IP system, it must be ensured that the client and server of the services used are compatible with each other.

1.7 Dokumentationen der Dienste

Für jeden Fachdienst wird eine eignene Dokumentation gepflegt. Die Version des Dokumentes entspricht der Version des Dienstes. Aktualisierungen und Änderungen werden in der Versionshistorie gepflegt. Über das Datum der Veröffentlichung im Titelblatt können Dokumente gleicher Version unterschieden werden.

Von den Diensten gemeinsam genutzte Datenstrukturen und Aufzählungstypen sind in der VDV 301-2-1 beschreiben.

Die freigegeben Dokumente der Spezifikationen sind auf der Website des VDV verfügbar (www.vdv.de/ip-kom-oev.aspx)

❖ Documentation of services

A separate documentation is maintained for each IBIS-IP service. The version of the document corresponds to the version of the service. Updates and changes are maintained in the version history. Documents of the same version can be distinguished by the date of publication in the title page.

Data structures and enumeration types shared by the services are described in VDV 301-2-1.

The released documents of the specifications are available on the VDV website (www.vdv.de/ip-kom-oev.aspx)

2 Verwendete Kommunikationsprotokolle

In diesem Kapitel wird beschrieben, unter welchen Gesichtspunkten in IBIS-IP entsprechende Kommunikationsprotokolle zum Einsatz kommen. Dazu folgt zunächst eine Grundlagendefinition der unteren Vermittlungsprotokolle. Anschließend folgt eine kurze Betrachtung des fachlichen Kontexts und der daraus folgenden Festlegung des sich in der Applikationsebene befindlichen Kommunikationsprotokolls. Zuletzt wird der Stand von IBIS-IP bzgl. anderer Kommunikationsprotokolle beschrieben.

❖ Communication Protocols used

This chapter describes, under which aspect in IBIS-IP respective communication protocols are used. First, the basics of the lower-level exchange protocols are defined. Next, a short discussion of the functional context and the resulting definition of the communication protocol located in the application layer are given. Finally, the status of IBIS-IP regarding other communication protocols is described.

2.1 Adressierung / Adressing

2.1.1 IP-Adressen

Die IP-Adressen werden in IBIS-IP dezentral mit Hilfe eines Teiles des „Zero Conf“ (vgl. RFC 3927 zur automatischen Adressvergabe) vergeben. Eventuelle feste Adressbereiche müssten projektspezifisch festgelegt werden. In Bezug auf Adressbereiche macht die RFC bereits Vorgaben (169.254.xxx.xxx), welche für ein interoperables Netzwerk beachtet werden müssen. Wichtig ist allerdings einzig, dass diese Adressbereiche konsistent zwischen allen Teilnehmern sein müssen.

Best-Practice-Hinweis:
Abweichend von obiger Festlegung hat es sich in der Praxis bewährt, fixe IP-Adressen oder Adressvergabe per DHCP zu verwenden.

❖ IP Addresses

The IP addresses are allocated decentralized in IBIS-IP using a part of "Zero Conf" (cf. RFC 2927 for automatic address allocation). Possibly fixed address ranges must be defined project-specifically. With respect to the address ranges, RFC already provides specifications (169.254.xxx.xxx), which must be observed for an interoperable network. However, the only important requirement is that these address ranges must be consistent among all participants.

Best Practice Note:
In contrast to the above definition, it has proven itself in practice to use fixed IP addresses or address assignment via DHCP.

2.1.2 Subnetzmasken/Gateways

Es ist projektspezifisch festzulegen, welche Adressen verwendet werden sollen.

❖ Subnet Masks/Gateways

The addresses to be used must be defined project-specifically.

2.2 Konfigurationsparameter für TCP und UDP

Eine spezielle Anpassung der Kommunikationsparameter der TCP- und UDP-Protokollen muss für IBIS-IP nicht erfolgen. Eine Festlegung der Ports ist insoweit nicht notwendig, als diese Informationen über DNS-SD (vgl. Kapitel 3.3.1) mitgeteilt werden. So kann jede Applikation bzw. jeder Gerätehersteller die Ports frei wählen. Es sollte im Sinne der Standardkonformität auf Ports im Bereich 0-1024 verzichtet werden (vgl. RFC 6335).

❖ Configuration Parameters for TCP and UDP

The communication parameters of the TCP and UDP protocols must not be specially adapted for IBIS-IP. The definition of a port is not required, as this information is communicated via DNS-SD (cf. Chapter 3.3.1). Every application and/or device manufacturer can freely select the ports. In terms of standard conformity, ports in the range of 0-1024 should be omitted (cf. RFC 6335).

2.3 Gültigkeitsdauer von Informationen

Wie in der VDV-Schrift 301-1 beschrieben, lassen sich die Informationen anhand ihrer Gültigkeitsdauer im Wesentlichen in zwei Klassen einteilen. Es gibt

— zyklische Informationen und

— ereignisgesteuerte Informationen.

Wird bei der Umsetzung einer Fachkomponente in Dienste (vgl. Kapitel 3) festgestellt, dass diese Fachkomponente sowohl zyklische als auch ereignisgesteuerte Information bereitstellt, so sind hierfür jeweils unterschiedliche Dienste zu erstellen.

❖ Information Validity Period

As described in VDV recommendation 301-1, the information can mainly be divided into two classes based on their validity period. There are

— periodic information, and

— event triggered information.

If, during the implementation of a functional component in the service (cf. chapter 3), it is determined that this functional component provides cyclic as well as event-controlled information, different services must be provided.

2.3.1 Zyklische Informationen

Es gibt Informationen,

— die sich zyklisch in kurzen Zeitabständen (~sekündlich) ändern bzw. ändern können und

— bei denen ein schneller und gleichzeitiger Transport der Information zu den Konsumenten wichtiger ist als eine gesicherte Informationsübertragung.

Dabei handelt es sich in der Regel um Informationen, die von Geräten oder Schnittstellen zyklisch in kurzen Zeitabständen zur Verfügung gestellt werden. Dazu gehören u. a.

— Uhrzeit (sekündlich)

— Odometer (<=sekündlich)

— GNSS-Koordinaten (<=sekündlich)

Zyklische Informationen ändern sich i. d. R. ohne eine Änderung des betrieblichen oder fachlichen Zustands eines Fahrzeugs zu bewirken. Sie werden i. d. R. von den Diensten der Basisfunktionen bereitgestellt.

Beispiel*:*
In einem Fahrzeug, das sich in Schrittgeschwindigkeit bewegt, ändern sich sekündlich die Odometerwerte, die GPS-Koordinaten und die Uhrzeit. Dennoch kommt es nur gelegentlich zu einer Änderung des betrieblichen oder fachlichen Zustands des Fahrzeugs, z. B. wenn eine neue Haltestelle erreicht wird.

❖ Periodic Information

There is information

— that changes or can change periodically in short intervals (~every second), and

— for which a fast and concurrent transport of information to the consumer is more important than reliable information transfer.

This is normally information that is provided by devices or interfaces in short time intervals. Among others, this includes

— Time (every second)

— Odometer (<=every second)

— GNSS coordinates (<=every second)

Periodic information normally changes without causing a change in the operating or functional state of a vehicle. It is normally provided by the services of the base functions.

Example:

In a vehicle that moves at walking speed, the odometer values, GPS coordinates, and the time change every second. However, the operating or functional state of the vehicle is changed occasionally only, e.g. when a new stop is reached.

2.3.2 Ereignisgesteuerte Informationen

Es gibt Informationen,

— die sich seltener als sekündlich und ereignisgesteuert ändern,

— die i. d. R. etwas über den fachlichen oder betrieblichen Zustand eines Fahrzeugs aussagen,

— bei denen sichergestellt sein muss, dass auch alle Konsumenten von dieser Änderung zuverlässig erfahren.

Beispiel*:*
Bei Erreichen einer neuen Haltestelle ändert sich der fachliche Zustand eines Fahrzeugs. Es soll sichergestellt sein, dass alle Dienste und Applikationen informiert werden, für die diese Änderung relevant ist, wie die Fahrgast-Innenanzeiger, die Fahrgast-Ansage, das Fahrer-MMI, die mobilen Kundenendgeräte, die Automatische Fahrgastzählung, das itcs, etc.

❖ Event triggered Information

There is information

— that changes less frequently than every second and in an event-driven manner,

— that normally provides information about the functional or operating state of a vehicle,

— for which it must be ensured that all consumers are reliably informed about this change.

Example:

When a new stop is reached, the functional state of a vehicle changes. It should be ensured that all services and applications are informed, for which this change is relevant, such as interior passenger displays, passenger announcement, driver MMI, mobile customer end devices, automatic passenger counting, itcs, etc.

2.4 Verwendung des UDP- und HTTP-Protokolls

Um den unterschiedlichen fachlichen Kontexten von zyklischen und ereignisgesteuerten Informationen Rechnung zu tragen, werden für IBIS-IP die folgenden Festlegungen getroffen:

— Für die Übertragung von sich zyklisch ändernden Daten wird das UDP-Protokoll verwendet.
Die Daten werden dazu in geeigneten zeitlichen Abständen an eine Multicast-Adresse gesendet.
Der entsprechende Dienst wird dann als UDP-Dienst bezeichnet.
Die Wahl der geeigneten zeitlichen Abstände richtet sich nach den fachlichen Anforderungen und den technischen Möglichkeiten der informationsliefernden Schnittstellen. Aufgrund der derzeitigen technischen Gegebenheiten ist eine Zykluszeit unter 1s für UDP-Dienste zu vermeiden.

— Für die Übertragung von Daten, die sich ereignisgesteuert ändern, wird das HTTP-Protokoll verwendet.
Der entsprechende Dienst wird dann als HTTP-Dienst bezeichnet.
Aufgrund dieser Entscheidung muss jedes Gerät, das einen HTTP-Dienst anbietet, die Funktionalität eines eigenen HTTP-Servers haben. Mit HTTP-Diensten steht für die Kommunikation zwischen zwei Teilnehmern eine Punkt-zu-Punkt-Verbindung zur Verfügung. Das Zustandekommen und die sichere Übertragung der Information werden hierbei durch den HTTP-Protokoll-Stack gewährleistet.

❖ Use of the UDP and HTTP Protocols

The following definitions are given for IBIS-IP in order to support the different functional contexts of periodic and event triggered information:

— The UDP protocol is used for transferring periodically changing data. The data is sent in suitable time intervals to a multicast address.
The respective service is then referred to as UDP service.
The time intervals are selected according to the functional requirements and the technical possibilities of the interfaces delivering information. Based on the current technical circumstances, a cycle time of less than 1 sec must be avoided for UDP services.

— The HTTP protocol is used for transferring data that changes in an event-driven manner.
The respective service is then referred to as HTTP service.

Based on this decision, every device that offers an HTTP service, must have the functionality of an own HTTP server. In the case of HTTP services, a point-to-point connection is available for the communication between two participants. Establishment and reliable information transfer are ensured by the HTTP protocol stack.

2.5 Weitere Protokolle

Für den Zeitserver im System wird das etablierte Standardprotokoll zur Zeitsynchronisation „SNTP“ verwendetet. Das Simple Network Time Protocol (SNTP) erlaubt es, auf einfache Art und Weise mithilfe von UDP die aktuelle Uhrzeit in einem Netzwerk zu verteilen. Weitere Informationen zur Umsetzung siehe VDV 301-2-10.

Bei der Umsetzung weiterer Fachkomponenten in Dienste ist zu erwarten, dass noch weitere IP-basierte Kommunikationsprotokolle zum Einsatz kommen (z. B. das RTP-Protokoll für das Streamen von Audio- und Videodaten). Darüber hinaus gehende Protokolle werden von der Version 1.0 von IBIS-IP nicht berücksichtigt.

❖ Other Protocols

The established standard protocol for time synchronization "SNTP" is used for the timeserver in the system. The Simple Network Time Protocol (SNTP) allows the distribution of the current time in a network in a simple manner using UDP. Further information regarding the implementation, see VDV 301-2-10.

During the implementation of other functional components in services, it is expected that further IP-based communication protocols will be used (e.g. RTP protocol for audio and video data streaming). Protocols going beyond that are not considered in version 1.0 of IBIS-IP.

3 Veröffentlichung und Kommunikation der Dienste

Applikationen und Dienste innerhalb eines IBIS-IP-Systems kennen

— den Dienstnamen und

— die IBIS-IP-Version

der Dienste, mit denen sie kommunizieren müssen.

Diese Eigenschaften sind Applikations- bzw. Dienst-inhärent, müssen also nicht konfiguriert werden.

Beispiel:
Die Applikation eines Fahrgast-Anzeigers benötigt den Dienst `CustomerInformationService` *(repräsentiert die Fahrgast-Informations-Ermittlung) in der Version X um Fahrgast-Informationsdaten anzeigen zu können. Mit Daten anderer Dienste oder Daten der Version X+1 kann der Fahrgast-Anzeiger nicht umgehen.*

Sie sind jedoch nicht ausreichend, um einen Dienst systemweit eindeutig zu adressieren (vgl. Kapitel 5.1.4.2.1) und damit nutzen zu können. Für eine systemweit eindeutige Findung des Dienstes ist es nötig zu wissen, auf welchem Gerät der erforderliche Dienst läuft. Wo und auf welchem Gerät ein welcher Dienst läuft, ist in der spezifischen Konfiguration eines Systems festgelegt, also nicht Applikations- oder Dienstinhärent.

❖ Service Publication and Communication

Applications and services within an IBIS-IP system know the

- service name, and
- IBIS-IP version

of the services, with which they must communicate.

These properties are application and/or service-inherent, i.e. must not be configured.

Example:

The application of a passenger display requires the `CustomerInformationService` *service (represents the Customer Information Determination) in version X, in order to be able to display the passenger information data. The passenger display cannot handle the data of other services or data of version X+1.*

However, they are not sufficient to uniquely address a service system-wide (cf. chapter5.1.4.2.1), and thus to use it. To uniquely find the service system-wide, the information, on which device the required service is running, is needed. Where and on which device which service is running is defined in the specific configuration of a system, i.e. it is not inherent to the application or service.

3.1 Von Fachkomponenten zu Diensten

Ein IBIS-IP-System besteht aus mehreren Fachkomponenten, die miteinander Daten austauschen. Eine Fachkomponente kann

— eine abstrakte Schnittstelle zu einem anderen System oder Gerät,

— eine Applikation,

— ein Dienst oder

— ein Gerät

sein (vgl. VDV-Schrift 301-1).

Alle Applikationen und Dienste eines IBIS-IP-Systems laufen auf Geräten, die miteinander per IP kommunizieren. Dabei stellen IBIS-IP-Applikationen und IBIS-IP-Dienste jeweils softwaretechnische Umsetzungen von Fachkomponenten dar.

Applikationen und Dienste unterscheiden sich dadurch, dass Applikationen im Sinne der IBIS-IP-Systemarchitektur reine Informationsempfänger sind, also keine Daten über eine IBIS-IP-Schnittstelle anderen Diensten oder Applikationen zur Verfügung stellen. Dienste hingegen bieten Schnittstellen zum Datenaustausch für andere Dienste und Applikationen des IBIS-IP-Systems an. Dazu bieten die Dienste eine Reihe von Operationen an. Mit Hilfe dieser Operationen können Daten mit den Diensten ausgetauscht werden oder Aktionen bei Diensten ausgelöst werden.

Jedes Gerät, das an einem IBIS-IP-System teilnimmt und bzgl. seines Status und seiner Erreichbarkeit vom IBIS-IP-System überwacht werden soll, verfügt mindestens über einen Dienst `DeviceManagementService`, der die Fachkomponente Gerätemanagement repräsentiert. Das Gerätemanagement stellt insbesondere Operationen zur Verfügung, mit deren Hilfe weitere Dienste auf einem Gerät gestartet werden können.

Der Austausch der Daten mit den Diensten erfolgt in IBIS-IP in Form von XML-Daten. Je nach fachlichem Kontext erfolgt der Austausch der XML-Datenstrukturen über das HTTP- oder das UDP-Protokoll.

Beispiel:

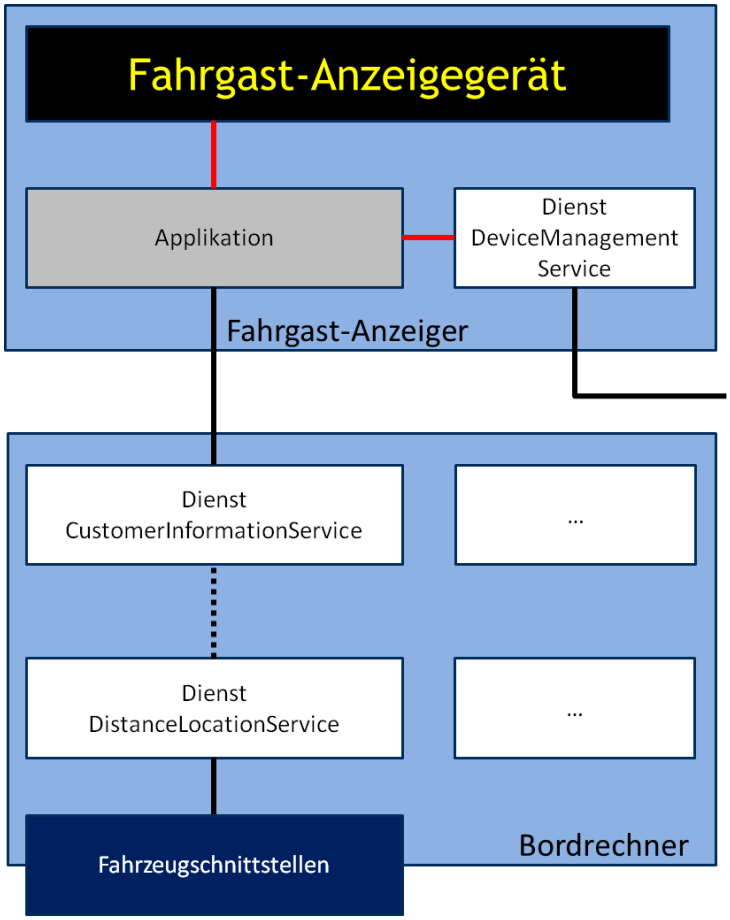

Abbildung 1 Beispiel zu den verschiedenen Repräsentationen von Fachkomponenten. Schwarze Linien: IBIS-IP, rote Linien: proprietär. Man beachte, dass die Fachkomponente Fahrgast-Anzeiger sowohl Geräte-, Applikations- wie auch Diensteigenschaften hat.

An dem System (vgl. Kapitel 1.4) sind zwei Geräte beteiligt

— ein Fahrgast-Anzeiger und

— ein Bordrechner

Auf dem Bordrechner laufen

— der Dienst `CustomerInformationService` (repräsentiert die Fahrgastinformationsermittlung) und

— der Dienst `DistanceLocationService` (repräsentiert einen Teil der physikalischen Ortung)

Der `DistanceLocationService` stellt als Dienst Daten zur Verfügung, die (über mehrere Zwischenschritte) dazu beitragen, die korrekten Informationen für die Fahrgast-Informations-Ermittlung zu ermitteln. Der `DistanceLocationService` bezieht dazu Daten (Odometer-Informationen, GNSS-Positionsdaten) von den Fahrzeugschnittstellen.

Der `CustomerInformationService` stellt als Dienst die jeweils aktuellen Fahrgastinformationen zur Verfügung.

Auf dem Fahrgast-Anzeiger läuft eine Applikation, die

— Daten vom Dienst `CustomerInformationService` abruft und

— über eine proprietäre Schnittstelle auf der Anzeige-Einheit des Fahrgast-Anzeigers entsprechend darstellt.

Die Applikation stellt anderen IBIS-IP-Fachkomponenten keine weiteren Informationen im Rahmen von IBIS-IP zur Verfügung, ist folglich kein Dienst.

Auf dem Fahrgast-Anzeiger läuft ein Dienst `DeviceManagementService`, über den eine Überwachung des Fahrgast-Anzeigers, im Sinne von Applikation und Gerät, durch das IBIS-IP-System erfolgen kann.

❖ From functional Components to Services

An IBIS-IP system consists of several functional components, which exchange data. A functional component can be

- an abstract interface to another system or device,
- an application,
- a service, or
- a device

(cf. VDV recommendation 301-1).

All applications and services of an IBIS-IP system run on devices, which communicate by IP. In this context, IBIS-IP applications and IBIS-IP services represent the software implementation of functional components.

Applications and services differ in that applications in terms of the IBIS-IP system architecture are pure information recipients, i.e. they do not provide data via an IBIS-IP interface to other services or applications. Services, on the other side, offer data exchange interfaces for other services and applications of the IBIS-IP system. The services offer a list of operations for this purpose. Using these operations, data can be exchanged with the services or actions triggered in the services.

Every device that participates in an IBIS-IP system and that should be monitored by the IBIS-IP system regarding its status and addressability features at least one `DeviceManagementService` service, which represents the device management functional component. Device management in particular provides operations, using which further services can be started on a device.

Data is exchanged with the services in IBIS-IP in the form of XML data. Depending on the functional context, the XML data structures are exchanged via the HTTP or UDP protocol.

Example:

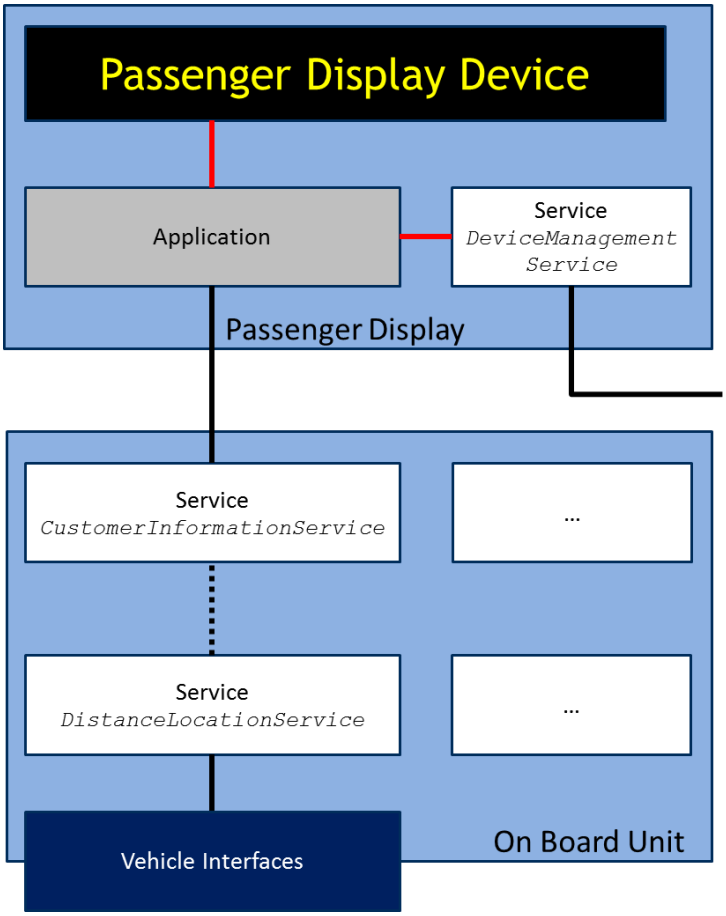

Figure 2: *Example for the different representations of a functional component. Black lines: IBIS-IP, red lines: proprietary. Please observe that the passenger display functional component has device properties, application properties, and service properties.*

Two devices participate in the system (cf. chapter 1.4):

- *passenger display, and*
- *on-board computer.*

The following services run on the on-board computer:

- `CustomerInformationService` *(represents the customer information determination), and*
- `DistanceLocationService` *(represents a part of physical locating).*

The `DistanceLocationService` *provides data (via several intermediate steps) that contributes to the determination of the correct information for the customer information determination. For this purpose, the* `DistanceLocationService` *gets data (odometer information, GNSS position data) from the vehicle interfaces.*

The `CustomerInformationService` *provides the respectively current passenger information.*

An application runs on the passenger display, which

- *retrieves data from the* `CustomerInformationService`*, and*
- *represents it via a proprietary interface on the display unit of the passenger display.*

The application does not provide any further information to other IBIS-IP functional components within the scope of IBIS-IP. Thus, it is not a service.

A `DeviceManagementService` *runs on the passenger display, using which the passenger display can be monitored by the IBIS-IP system in terms of the application and the device.*

3.2 Spezifizierte Dienste

Eine Übersicht der aktuellen Dienste wird durch den VDV veröffentlicht und ist auf der Website verfügbar: www.vdv.de/ip-kom-oev.aspx

❖ Specified Services

An overview of the current services is published by VDV and is available on the website: www.vdv.de/ip-kom-oev.aspx

3.3 Veröffentlichung via DNS-SD

Für die Verwendung und Verbreitung von Informationen zu Diensten wird bei diesem Verfahren auf bekannte Standardmechanismen der Netzwerktechnologie zurückgegriffen. Mit Hilfe von sogenannter „SRV-Records“ (vgl. Kapitel 3.3.1) und „TXT-Records“ (vgl. Kapitel 3.3.2) wird allgemein das Kommunikationsverfahren unter der Bezeichnung DNS-SD (Domain Name System – Service Discovery) zusammengefasst. Es gibt für verschiedene Betriebssysteme kostenfreie Bibliotheken [6], welche das Arbeiten mit dieser Technologie ermöglichen. Bei Verwendung dieser Bibliotheken ist es i. d. R. für den Nutzer transparent, welche Records für die Übertragung welcher Information verwendet werden.

❖ Publication via DNS-SD

Known standard network technology mechanisms are used in this method for information use and distribution to the services. Using so-called SRV records (cf. chapter 3.3.1) and TXT records (cf. chapter 3.3.2), the communication method is generally summarized under the name DNS-SD (Domain Name System – Service Discovery). Free-of-charge libraries [6] are available for the different operating systems, which allow working with this technology. When these libraries are used, it is normally transparent to the user, which records are used for transferring which data.

3.3.1 Nutzung des SRV-Records

Aufbauend auf das standardisierte IP-Kommunikationsprotokoll UDP und den Mechanismen, mit denen in IP-basierten Netzen Informationen über Rechnernamen ausgetauscht werden (sogenannten DNS-Records), wurde im RFC 2782 von der Internet Engineering Task Force (IETF) eine Erweiterung dieses Namensaustauschverfahrens speziell für die Bekanntgabe von Diensten spezifiziert.

Mit dieser standardisierten Erweiterung der DNS-Records, den sogenannten SRV-Records, ist es möglich, dass innerhalb von IP-Netzen von den Geräten automatisch bekanntgegeben wird, welche Dienste auf diesen Geräten angeboten werden und wie diese Dienste angesprochen werden können.

Der Aufbau eines solchen SRV-Records folgt dem Muster

```
<ServiceProtoName
  TTL  Class  SRV  Priority  Weight  Port  Target>
```

und sieht exemplarisch wie folgt aus

```
<CustomerInformationService._ibisip_http._tcp.local
  3600  IN  SRV  10  0  389  OnboardUnit_1.local>
```

Die Bedeutung der Felder ist nachfolgend beschrieben.

Datenfeld (nach RFC 2782)	**Beispielwert**	**Beschreibung**
`Service`	`CustomerInformationService.` (mit . am Ende)	Dienstname (mögliche Werte für Dienste in IBIS-IP finden sich in Kapitel Kapitel 3.2). Man beachte den nachfolgenden Hinweis bezüglich des eindeutigen Namens.
`Proto`	`_ibisip_http._tcp.` (mit . am Ende)	Kommunikationsprotokoll, in IBIS-IP gibt es nur zwei mögliche Werte:

Datenfeld (nach RFC 2782)	Beispielwert	Beschreibung
		`_ibisip_udp._udp` und `_ibisip_http._tcp` (vgl. auch 2.2, 3.4 und 3.5)
`Name`	`local`	die Domäne des Dienstes
`TTL (time to live)`	`3600`	Gültigkeitsdauer des Records (in Sekunden)
`Class`	`IN`	Klassenbegriff entsprechend RFC 1035, SRV-Records gehören in die `IN`-Klasse, `IN` steht hierbei für Internet
`SRV`	`SRV`	Art des Records, hier also ein `SRV`-Record
`Priority`	`10`	Priorität für den Inhalt (hier also der Dienst) gibt es zwei Dienste gleichen Namens, dann wird der Dienst mit geringerer Priorität bevorzugt
`Weight`	`0`	Gewicht für den Inhalt (hier also der Dienst), bei gleichem Gewicht wird der Dienst mit dem geringeren Gewicht bevorzugt
`Port`	`389`	gibt an, unter welchem Port der bekanntgegebene Dienst angesprochen werden kann
`Target`	`OnboardUnit_1`	DNS-Name oder IP-Adresse des Zielhostrechners, auf dem der Dienst läuft

Tabelle 3 Bedeutungen der SRV-Records in DNS-SD

Best-Practice-Hinweis bezüglich der Vermeidung doppelter Service-Namen

Gemäß DNS-SD muss jeder Service-Name eindeutig sein. Manche DNS-SD-Implementierungen haben Mühe mit gleichen Service-Namen und ergänzen einen Postfix am Ende des Service-Namens. Dies kann aber zu ungewünschtem Aufstartverhalten führen. Daher wird vorgeschlagen, einen Postfix anzuhängen, um den Service-Namen eindeutig zu halten. Der Postfix soll aus Hersteller, Gerätebezeichnung und sofern notwendig der Device-ID bestehen. Als Trennzeichen soll ein „_“ verwendet werden. Es wird empfohlen, denselben Postfix beim Fachdienst wie auch beim entsprechenden DeviceManagementService anzuhängen. Somit kann bei Systemen welche mehrere DeviceManagementService auf einem physikalischen Gerät installiert haben, eine Zuordnung zwischen Fachdienst und DeviceManagementService gemacht werden.

Beispiel:
`CustomerInformationService_mySupplier_myDeviceName`
`DeviceManagementService_yourSupplier_hisDeviceName_1`

❖

❖ Use of SRV Records

Based on the standardized IP communication protocol UDP and the mechanisms, using which information is exchanged about computer names (so-called DNS records) in IP-based networks, an extension of this name exchange method was specified by the Internet Engineering Task Force (IETF) in RFC 2782 specifically for the publication of services.

Using this standardized extension of the DNS records, the so-called SRV records, it can be automatically announced within IP networks of the devices, which services are offered on these devices, and how these services can be addressed.

The structure of such SRV records follows the pattern

```
<_Service._Proto.Name TTL Class SRV Priority Weight Port Target>.
```

The following is an example

```
<CustomerInformationService._ibisip_http._tcp.local
  3600  IN  SRV  10  0  389  OnboardUnit_1.local>
```

The meaning of the fields is described below.

Data field (acc. to RFC 2782)	**Example value**	**Description**
`Service`	`CustomerInformationService.` (with `.` at the end)	Service name (possible values for services in IBIS-IP can be found in chapter 3.2)
`Proto`	`_ibisip_http._tcp.` (with `.` at the end)	Communication protocol, there are two values only in IBIS-IP: `_ibisip_udp._udp` and `_ibisip_http._tcp` (compare also to 2.2, 3.4 and 3.5)
`Name`	`local.` (with `.` at the end)	Domain of the service
`TTL (time to live)`	`3600`	Validity period of the record (in seconds)
`Class`	`IN`	Class according to RFC 1035, SRV records belong to the `IN` class, `IN` stands for Internet
`SRV`	`SRV`	Type of the record, i.e. here a `SRV` record
`Priority`	`10`	Priority for the content (here the service). If there are two services with the same name, the service with lower priority is preferred
`Weight`	`0`	Weight for the content (here the service). If there are two services with the same name, the service with the lower weight is preferred
`Port`	`389`	Indicates, on which port the announced service can be accessed
`Target`	`OnboardUnit_1.` (with `.` at the end)	DNS name or IP address of the destination host computer, where the service runs

Table 4: Meanings of the SRV record in DNS-SD

Best practice advice on avoiding duplicate service names

According to DNS-SD, each service name must be unique. Some DNS SD implementations have trouble with identical service names and add a postfix at the end of the service name. However, this can lead to undesired start-up behavior. It is therefore suggested that you attach a postfix to keep the service name unique. The Postfix should consist of manufacturer, device name and, if necessary, the device ID. A "_" is to be used as separator character. It is recommended that you attach the same postfix to the specialized service as to the corresponding DeviceManagementService. Thus, for systems that have several DeviceManagementService installed on one physical device, an assignment can be made between specialist service and DeviceManagementService.

Example:
`CustomerInformationService_mySupplier_myDeviceName`
`DeviceManagementService_yourSupplier_hisDeviceName_1`

3.3.2 Nutzung des TXT-Records

Da nicht alle für IBIS-IP relevanten Informationen für die Dienstidentifikation im Standard SRV-Record untergebracht werden können, werden weitere Informationen mit Hilfe von sogenannten TXT-Records bekannt gegeben (vgl. RFC 1464).

Der TXT-Record erlaubt die Übergabe beliebiger Attribut-Werte-Paare in der Form

`<Attribut Name>=<Attribut Wert>.`

In IBIS-IP werden folgende Attribute verwendet:

Attribut	**Optional/ Pflicht**	**Beispielwert**	**Beschreibung**
`Ver`	Pflicht für alle IBIS-IP-Dienste	`1.0`	IBIS-IP-Version des Dienstes (vgl. Kapitel 1.6)
`Path`	Optional	`testversion_1.1`	Pfadinformation, falls sie vorhanden ist, muss sie zusätzlich zu den Angaben Port und Zielhost aus dem SRV-Record bei der techn. Adressierung eines Dienstes berücksichtigt werden (vgl. Kapitel 5.1.4.2ff, vgl. auch Beispiel. in Kapitel 3.5)

Attribut	Optional/ Pflicht	Beispielwert	Beschreibung
`multicast`	Pflicht für UDP-Dienste	`239.0.0.1`	Angabe der Multicast-Adresse, mit der die Daten des UDP-Dienstes veröffentlicht werden.
`sntp-server`	Pflicht für den Dienst zur Zeitsynchronisation	`192.168.0.22`	Angabe der IP-Adresse unter welcher der SNTP-Server im IBIS-IP-Netzwerk zu finden ist.

Tabelle 5 Bedeutungen der TXT-Records in DNS-SD (inklusive Festlegungen für IBIS-IP)

❖ Use of TXT Records

As not all information for the service identification relevant to IBIS-IP can be accommodated in the standard SRV record, further information is announced using so-called TXT records (cf. RFC 1464).

The TXT record allows to provide any attribute-value pairs in the form

`<Attribute Name>=<Attribute Value>.`

The following attributes are used in IBIS-IP:

Attribute	Optional/ Mandatory	Example value	Description
`ver`	Mandatory for all IBIS-IP services	`1.0`	IBIS-IP version of the service (cf. chapter 1.6)
`path`	Optional	`testversion_1.1`	Path information, if available, must be considered in addition to port and destination host from the SRV record for technical addressing of a service (cf. chapter 5.1.4.2pp, compare also to the example in chapter 3.5)
`multicast`	Mandatory for UDP services	`239.0.0.1`	Multicast address, using which the UDP service data is published.
`sntp-server`	Mandatory for the time synchronization service	`192.168.0.22`	IP address, under which the SNTP server can be found in the IBIS-IP network.

Table 6: Meanings of the TXT record in DNS-SD (including the definitions for IBIS-IP)

3.4 Veröffentlichung von UDP-Diensten

UDP-Dienste geben bei Ihrer Veröffentlichung via DNS-SD mindestens folgende dienstspezifische Informationen an:

Im SRV-Record:

— Dienstname

— Protokoll = `_ibisip_udp`

Im TXT-Record:

— `ver`

— `multicast`

Um Informationen der UDP-Dienste zu bekommen, muss der Client der bekanntgegebenen Multicast-Gruppe beitreten. Dort werden von den Diensten die entsprechenden Nachrichten per UDP-Telegramm verbreitet. Mit dem Beitritt zur Multicast-Gruppe können die entsprechenden Telegramme empfangen werden.

Eine explizite Abfrage von Informationen oder auch andere Mechanismen, welche bei den HTTP-Diensten verfügbar sind (siehe Kapitel 2.4), sind in dieser Form nicht vorgesehen.

❖ Publication of UDP Services

UDP services indicate at least the following service-specific information during their publication via DNS-SD:

In the SRV record:

- Service name
- Protocol = `_ibisip_udp`

In the TXT record:

- `ver`
- `multicast`

In order to receive information of the UDP services, the client must join the announcing multicast group. There, the respective messages are distributed by the services via UDP telegram. After joining the multicast group, the respective telegrams can be received.

An explicit information request or other mechanisms available for HTTP services (cf. chapter2.4) are not planned.

3.5 Veröffentlichung von HTTP-Diensten

HTTP-Dienste geben bei Ihrer Veröffentlichung via DNS-SD mindestens folgende dienstspezifische Informationen an:

im SRV-Record:

— Dienstname

— Protokoll = `_ibisip_http`

— Port

— Zielhost

Im TXT-Record:

— `ver` (Pflicht)

— `path` (optional)

Beispiel:
Um beispielsweise auf einem Gerät namens „tst01“ unter Port 1234 die Operation ***`GetDeviceConfiguration`*** *des* `DeviceManagementService` *auszuführen, ist die dafür maßgebliche Adresse für die HTTP-Kommunikation*

```
<tst01:1234/DeviceManagementService/GetDeviceConfiguration>
```

Ist zusätzlich im TXT-Record dieses Dienstes ein optionaler „`path=IBIS-IP`“ eingetragen, so lautet die anzusprechende Adresse

```
<tst01:1234/IBIS-
IP/DeviceManagementService/GetDeviceConfiguration>
```

❖ Publication of HTTP Services

HTPP services indicate <u>at least</u> the following service-specific information during their publication via DNS-SD:

In the SRV record:

- Service name
- Protocol = `_ibisip_http`
- Port
- Destination host

In the TXT record:

- `ver` (mandatory)
- `path` (optional)

Example:

In order to execute operation ***`GetDeviceConfiguration`*** *of the* `DeviceManagementService` *on a device called "tst01", the decisive address for the HTTP communication is*

```
<tst01:1234/DeviceManagementService/GetDeviceConfiguration>
```

If an optional "`path=IBIS-IP`" is additionally entered in the TXT record of this service, the address to be addressed is

```
<tst01:1234/IBIS-
IP/DeviceManagementService/GetDeviceConfiguration>
```

4 Konventionen für Dienste und Datenstrukturen

In dieser Schrift wird ausschließlich die Kommunikation mit den IBIS-IP-Diensten beschrieben, also insbesondere

— die Strukturierung der Daten, die in IBIS-IP ausgetauscht werden,

— welche Daten(Strukturen) von den Operationen als Ein- und Ausgabeparameter verwendet werden (Request-/Responsestrukturen) und

— welche Operationen durch welche Dienste bereitgestellt werden

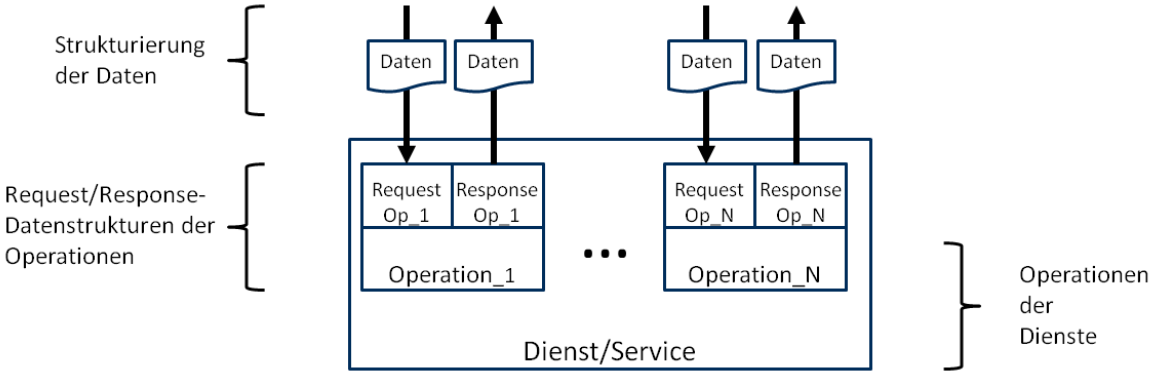

Abbildung 3 Darstellung der Strukturierung von Daten in Diensten

Dabei erfolgt sowohl eine Darstellung allgemeiner Prinzipien der Dienste-Kommunikation als auch die Beschreibung konkreter Umsetzungen von Fachkomponenten in Diensten und die Spezifikation der Basisdienste.

Geräte werden dabei berücksichtigt,

— weil sie für den Kontext, in dem ein Dienst läuft, relevant sind und

— weil sie durch das IBIS-IP-System überwacht werden.

Diese Schrift behandelt **nicht**

— die Umsetzung von Fachkomponenten in Form einer Applikation und

— die abstrakten Schnittstellen zu andern Systemen und Geräten

— die Spezifikation der spezifischen Dienstefür die Anwendungen

❖ Conventions for Services and Data Structures

This document exclusively describes the communication with IBIS IP services, i.e. in particular

- Structuring of data exchanged in IBIS-IP,
- Data (structures) used by operations as input and output parameters (request/response structures), and
- Operations provided by the different services.

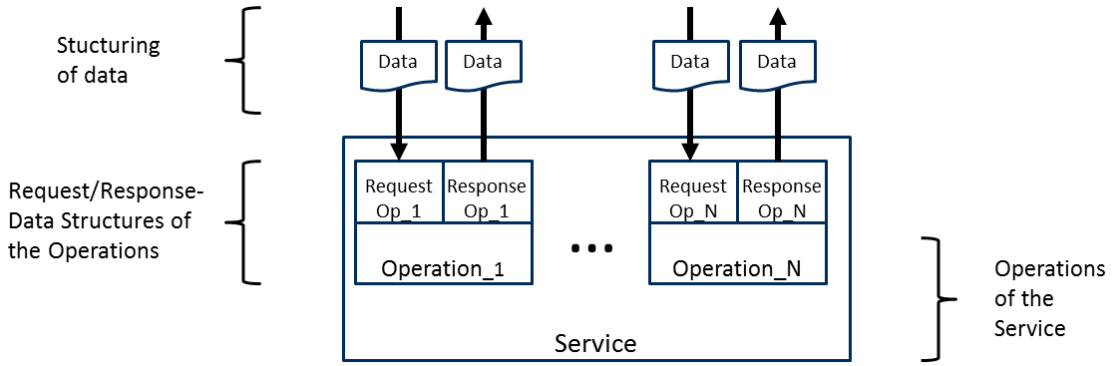

Figure 4: Representation of the structuring of data in services

In this context, the general principles of service communication are presented, and the concrete implementation of functional components in the services is described.

Devices are considered,

- As they are relevant to the context, in which the service runs, and
- As they are monitored by the IBIS-IP system.

This document does **not discuss the**

- Functional component implementation in the form of an application, and
- Abstract interfaces to other systems and devices.

4.1 Konventionen für HTTP-Dienste

HTTP-Dienste in IBIS-IP stellen ihre Funktionalität über Operationen zur Verfügung. Die Operationen dienen dazu, Aktionen beim Dienst auszulösen, die bestimmte Objekte betreffen, die der Dienst verwaltet.

❖ Conventions for HTTP Services

HTTP services in IBIS-IP provide their functionality via operations.
The operations are used to trigger actions for the service, which concern certain objects that are managed by the service.

4.1.1 Verwendung von HTTP-POST und HTTP-GET

IBIS-IP-Dienste bieten entweder Operationen an, denen

— Daten übergeben werden. In diesem Fall erfolgt der Operationsaufruf immer via HTTP-POST

— keine Daten übergeben werden. In diesem Fall erfolgt der Operationsaufruf immer via HTTP-GET.

❖ Use of HTTP-POST and HTTP-GET

IBIS-IP services offer operations,

a) To which data is provided. In this case, the operation is called via HTTP-POST.

b) To which no data is provided. In this case, the operation is called via HTTP-GET.

4.1.1.1 Operationsaufrufe mit HTTP-POST

Operationsaufrufe mit HTTP-POST werden immer dann verwendet, wenn dem Dienst Daten übergeben werden sollen.

Dabei ist durch den Dienst festgelegt, welche Datentypen bzw. Datenstrukturtypen vom Dienst akzeptiert und interpretiert werden können.

Beispiel:
Mit Hilfe der Operation ***`SetBlockNumber`*** *des Dienstes* `JourneyInformationService`*, kann ein neuer Umlauf beim* `JourneyInformationService` *eingestellt werden. Dazu wird dem* `JourneyInformationService` *eine entsprechende BlockNumber-Datenstruktur als Parameter übergeben.*

Der Aufruf erfolgt als HTTP-POST.

❖ Operation Calls with HTTP-POST

Operation calls with HTTP-POST are always used if data should be provided to the service.

In this context, the service defines which data types and/or data structure types can be accepted and interpreted by the service.

Example:

Using the ***`SetBlockNumber`*** *operation of the* `JourneyInformationService` *service, a new block can be adjusted at the* `JourneyInformationService`*. For this purpose, a respective BlockNumber data structure is provided to the* `JourneyInformationService`*.*

The call is realized as HTTP-POST.

4.1.1.2 Operationsaufrufe mit HTTP-GET

Operationsaufrufe mit HTTP-GET werden immer dann verwendet, wenn keine Übergabe von Daten an den Dienst erfolgt, d. h. wenn es sich um eine reine Abfrage von Daten handelt.

Sämtliche HTTP-GET Operationen können insbesondere über einen passenden Adress-Eintrag mit einem Browser abgerufen werden.

Beispiel:
Mit Hilfe der Operation ***`GetDeviceStatus`*** *des Dienstes* `DeviceManagementService`*, wird der Status des Geräts, auf dem der* `DeviceManagementService` *läuft, ermittelt. Ein Parameter ist bei dieser Abfrage nicht erforderlich.*

Der Aufruf erfolgt als HTTP-GET.

Die Abfrage kann mit einem Browser durch Eingabe der Adresszeile
`[…] DeviceManagementService/GetDeviceStatus`
erfolgen.

Es gilt insbesondere die Regel, dass für sämtliche **`Get`**-Operationen ein HTTP-GET verwendet wird.

❖ Operation Calls with HTTP-GET

Operation calls with HTTP=GET are always used, if no data is provided to the service, i.e. if the operation is a pure data request.

In particular, all HTTP-GET operations can be called using a browser via a suitable address entry.

Example:

Using the ***`GetDeviceStatus`*** *operation of the* `DeviceManagementService` *service, the status of the device is determined, on which* `DeviceManagementService` *is running. No parameters are required for this request.*

The call is realized as HTTP-GET.

The request can be realized using a browser by entering address line
`[…] DeviceManagementService/GetDeviceStatus`.

In particular an HTTP-GET must be used for all **`Get`** operations.

4.1.2 Namenskonventionen für Operationen

Operationsnamen enthalten mindestens ein Verb, das die Aktion beschreibt und ein Objekt, auf das sich die Aktion bezieht.

Folgende Verben finden bislang in unterschiedlichen Bereichen Verwendung:

Bei rein datenorientierten Operationen, sind dies die Verben

— Set,

— Retrieve,

— Get,

— List,

— Subscribe und

— Unsubscribe.

Bei Steuerungsoperationen, insb. des `DeviceManagementService`, sind dies die Verben

— Start,

— Stop,

— Restart,

— Activate und

— Deactivate.

Für Validierungsoperationen ist es zusätzlich das Verb

— Validate.

Um zu verhindern, dass gleiche Verben für verschiedene Aktionen oder verschiedene Verben für gleiche Aktionen verwendet werden, wurden kurze Konventionen für den Einsatz der Verben erarbeitet.

Die Konstruktion der Dienstnamen und die Konventionen zur Verwendung der Verben werden nachfolgend erläutert.

❖ Naming Conventions for Operations

Operation names contain at least one verb describing the action, and one object, to which the action refers.

The following verbs have been used in different areas:

In the case of purely data-oriented operations, the verbs are:

- Set,
- Retrieve,
- Get,
- List,
- Subscribe, and
- Unsubscribe.

In the case of control operations, in particular of the `DeviceManagementService`, the verbs are

- Start,
- Stop,
- Restart,
- Activate, and
- Deactivate.

For validation operations, an additional verb is

- Validate.

In order to prevent that the same verbs are used for different actions or different verbs for the same action, brief conventions were developed for the use of the verbs.

The service name construction and the conventions for using the verbs are explained below.

4.1.2.1 Rein datenorientierte Operationen

Sofern es sich bei den Objekten um Daten bzw. Datenstrukturen des Dienstes handelt, gilt Operationsname = Verb + betreffende Datenstruktur

Beispiel*:*
Die Operation zur Abfrage der Gerätekonfiguration heißt
`GetDeviceConfiguration` *= Get (Verb) + DeviceConfiguration (betreffende Datenstruktur).*

Die Daten können alle Arten von fachlichen Daten sein, z. B.

— Konfigurationsdaten,

— Ortungsdaten,

— Fahrgastinformationsdaten,

— Statusinformationen,

— etc.

Die datenorientierte Aktionen bezwecken in der Regel

— das Übergeben von Daten bzw. Datenstrukturen ODER

— das einmalige Abfragen von Daten bzw. Datenstrukturen ODER

— das Abonnieren auf Daten bzw. Datenstrukturen ODER

— das Beenden eines Abonnements auf Daten bzw. Datenstrukturen.

Die einzelnen Aktionen werden nachfolgend näher beschrieben.

❖ Purely data-oriented Operations

If the objects are data and/or data structures of the service, the following applies

Operation name = Verb + Relevant data structure

Example:
The operation for requesting the device configuration is
`GetDeviceConfiguration` *= Get (verb) + DeviceConfiguration (relevant data structure).*

The data can be functional data of any type, e.g.

- Configuration data,
- Location data,
- Passenger information data,
- Status information,
- etc.

The purpose of data-oriented actions is normally the

- Provision of data and/or data structures OR
- Single request of data and/or data structures OR
- Subscription to data and/or data structures OR
- Termination of a subscription to data and/or data structures.

The individual actions will be described in detail below.

4.1.2.1.1 Einstellen von Werten bei einem Dienst

Konvention:

Operationen, die dazu dienen, einen dienstinternen Wert festzulegen, beginnen mit dem Verb

Set (setzen, einstellen).

Der Name einer solchen Übergabe-Operation setzt sich außerdem aus einer Beschreibung des Daten- bzw. Datenstrukturobjekts zusammen, das von der Übergabe betroffen ist.

Beispiel:
Die Operation zur Einstellung der Gerätekonfiguration heißt
`SetDeviceConfiguration`

❖ Value Adjustment for a Service

Convention:

Operations used to define a service-internal value start with verb

Set.

The name of such an operation also consists of a description of the data and/or data structure object, which is to be defined.

Example:
The operation for setting (adjusting) the device configuration is
`SetDeviceConfiguration`

4.1.2.1.2 Einmalige Abfrage von aktuell gültigen Daten bzw. Datenstrukturen

Konvention:
Operationen, die der Abfrage von aktuell gültigen, dienstinternen Daten bzw. Datenstrukturen dienen, beginnen mit dem Verb

Get (erhalten).

Der Name einer solchen Abfrage-Operation setzt sich außerdem aus einer Beschreibung des Daten- bzw. Datenstrukturobjekts zusammen, das abgefragt wird.

Beispiel:
Die Operation zur Abfrage des Gerätestatus heißt
`GetDeviceStatus`

❖ Single Request of currently valid Data and/or Data Structures

Convention:

Operations used to request currently valid, service-internal data and/or data structures start with verb

Get.

The name of such a request operation also consists of a description of the data and/or data structure object, which is requested.

Example:
The operation for requesting the device status is
`GetDeviceStatus`

4.1.2.1.3 Abonnieren auf Daten bzw. Datenstrukturen

Damit ein Dienstnutzer von Änderungen einer Datenstruktur eines Dienstes erfährt, kann der Dienstnutzer den Dienst zyklisch mit Hilfe der Get-Abfragen abfragen. Solange sich der Wert der Datenstruktur nicht ändert, werden dabei unnötig Daten ausgetauscht. Daher ist in IBIS-IP durch die Verwendung von Abonnements ein solches Verfahren zu vermeiden.

Durch ein Abonnement auf eine Datenstruktur eines Dienstes wird der Abonnent automatisch vom Dienst über eine Änderung der Datenstruktur informiert.Konvention:

Operationen, die der Einrichtung eines Abonnements dienen, beginnen mit dem Verb

Subscribe (abonnieren, anmelden).

Beispiel:
Die Operation zur Einrichtung eines Abonnements auf den Status eines Gerätes heißt
`SubscribeDeviceStatus`

❖ Subscription to Data and/or Data Structures

To ensure that a service user is informed about changes to a data structure of a service, the service user can periodically request the service using Get requests. As long as the value of the data structure does not change, data is unnecessarily exchanged. For this reason, such a method is to be avoided in IBIS-IP by using subscriptions.

With the subscription to a data structure of a service, the subscriber is automatically informed by the service about a data structure change.

Convention:

Operations used for setting up the subscription start with verb

Subscribe.

Example:
The operation for setting up a subscription to the status of a device is
`SubscribeDeviceStatus`

4.1.2.1.4 Beenden eines Abonnements auf Daten bzw. Datenstrukturen

Um ein eingerichtetes Abonnement zu beenden, bedarf es einer weiteren Operation.

Konvention:

Operationen, die der Beendigung eines Abonnements dienen, beginnen mit dem Verb

Unsubscribe (abbestellen, abmelden).

Beispiel:
Die Operation zur Beendigung eines Abonnements auf den Status eines Gerätes heißt
`UnsubscribeDeviceStatus`

❖ Termination of a Subscription to Data and/or Data Structures

In order to terminate the set up subscription, another operation is required.

Convention:

Operations used for canceling the subscription start with verb

Unsubscribe.

Example:
The operation for canceling a subscription to the status of a device is
`UnsubscribeDeviceStatus`

4.1.2.1.5 Operationen zur Abfrage von spezifischen Informationen

Dienste können nicht nur Operationen bereitstellen, mit denen man auf die Werte aktuell gültiger Daten bzw. Datenstrukturen des Dienstes zugreifen kann, sondern auch Operationen, mit denen weitere spezifische Daten ermittelt werden, die nicht Teil des aktuell gültigen Datenstandes sind. Ein Abonnement auf derartige Daten ist nicht sinnvoll und ist deshalb nicht vorgesehen.

Konvention:

Operationen, die dazu dienen, spezifische Informationen zu ermitteln, die von einem übergebenen Parameter abhängen, beginnen mit dem Verb

Retrieve (abfragen, abrufen)

Wird kein Parameter übergeben, so wird eine Liste aller passenden Informationen in der Antwortstruktur zurückgegeben.

Beispiele:
Die Operation zur Ermittlung aller Informationen zu einem bestimmten Umlauf heißt
`RetrieveSpecificBlockInformation.`

Die Operation zur Ermittlung aller (IBIS-IP verlangt die letzten 10) Logdaten-Einträge heißt
`RetrieveLogMessages`

❖ Operations for requesting specific Information

Services cannot only provide operations, using which access is possible to the values of currently valid data and/or data structures of the service, but also operations, using which further specific data is determined, which is not part of the currently valid data version. Such subscriptions to data are not meaningful, and are thus not planned.

Convention:

Operations used to determine specific information that depend on a provided parameter, start with verb

Retrieve

If no parameter is provided, a list of all matching information is returned in the response structure.

Examples:
The operation for determining all information for a certain block is
`RetrieveSpecificBlockInformation.`

The operation for determining all log data entries (IBIS requires the last 10) is
`RetrieveLogMessages`

4.1.2.1.6 Operationen zur Abfrage von Datenlisten aus der Grunddatenversorgung

Konvention:

Operationen, die dazu dienen, alle im System vorhandenen Elemente einer bestimmten Information zu ermitteln, beginnen mit dem Verb

List (auflisten)

Der Name einer solchen Abfrage-Operation setzt sich außerdem aus einer Beschreibung des Daten- bzw. Datenstrukturobjekts zusammen, das abgefragt wird.

Beispiel:
Die Operation, um alle dem System bekannten Routen zu erhalten heißt
`ListAllRoutes`

Da bei diesen Operationen mit der Anfrage keine Parameter übergeben werden können, gelten die in Kapitel 4.1.1.2 dargestellten Möglichkeiten.

❖ Operations for requesting Data Lists from the basic Data Supply

Convention:

Operations used to determine all elements of a certain information available in the system, start with verb

List

The name of such a request operation also consists of a description of the data and/or data structure object, which is requested.

Example:
The operation to obtain all routes known to the system is
`ListAllRoutes`

As no parameters can be provided with the request for these operations, the possibilities presented in chapter 4.1.1.2 apply.

4.1.2.2 Steuerungsoperationen

Neben den in Kapitel 4.1.1 beschriebenen datenorientierten Operationen gibt es Operationen, bei denen das Objekt der Operation nicht Daten oder Datenstrukturen des Dienstes sind, sondern mit denen das Verhalten eines Dienstes oder Geräts gesteuert wird. Dies ist insbesondere beim Gerätemanagement der Fall.

Beim Gerätemanagement können über Operationen Dienste auf dem Gerät oder gar das gesamte Geräte gestoppt bzw. neu gestartet werden. Das Objekt ist in diesem Fall das Gerät (Device) oder der Dienst (Service).

Die Namensgebung der Dienste folgt dabei dem oben beschriebenen Muster aus Verb und Objekt.

❖ Control Operations

In addition to the data-oriented operations described in chapter 4.1.1, there are operations, where the object of the operation is not data or data structures, but using which the behavior of a service or device is controlled. This is particularly the case or device management.

In the case of device management, services can be stopped and/or restarted on the device using operations, or even the complete device. In this case, the object is the device or the service.

Object naming follows the pattern (verb and object) described above.

4.1.2.2.1 Starten

Operationen, die dem Starten eines Prozesses dienen, beginnen mit dem Verb

Start

Beispiel:
Die Operation zum Start eines Dienstes auf einem Gerät heißt
`StartService`.

❖ Start

Operations used for starting a process start with verb

Start

Example: *The operation for starting a service on a device is* ***`StartService`***.

4.1.2.2.2 Neustart auslösen

Operationen, die dem Neustart eines Prozesses dienen, beginnen mit dem Verb

Restart

Beispiel: *Die Operation, mit dem der Neustart eines Gerätes ausgelöst wird, heißt* ***`RestartDevice`***.

❖ Trigger Restart

Operations used for restarting a process start with verb

Restart

Example: *The operation for triggering a device restart is* ***`RestartDevice`***.

4.1.2.2.3 Stoppen

Operationen, die dem Stoppen eines Prozesses dienen, beginnen mit dem Verb

Stop

Beispiel: *Die Operation zum Stoppen eines Dienstes auf einem Gerät heißt* ***`StopService`***.

❖ Stop

Operations used for stopping a process start with verb

Stop

Example: *The operation for stopping a service on a device is* ***`StopService`***.

4.1.2.2.4 Deaktivieren eines Gerätes

Operationen, die dem Deaktivieren eines Geräts dienen, beginnen mit dem Verb

Deactivate

Beispiel:
Die Operation zum Deaktivieren eines Geräts heißt
`DeactivateDevice`

Ein über „Deactivate“ deaktiviertes Gerät kann nur durch Aufruf der Activate-Operation wieder in seinen vorherigen Betriebszustand zurückkehren.

❖ Device Deactivation

Operations used for deactivating a device start with verb

Deactivate

Example:
The operation for deactivating a device is
`DeactivateDevice`

A device deactivated via "Deactivate" can only return into is previous operating state by calling the Activate operation.

4.1.2.2.5 Aktivieren eines Gerätes

Operationen, die dem Aktivieren eines Geräts dienen, beginnen mit dem Verb

Activate

Beispiel:
Die Operation zum Aktivieren eines Geräts heißt
`ActivateDevice`

❖ Device Activation

Operations used for activating a device start with verb

Activate

Example:
The operation for activating a device is
`ActivateDevice`

4.1.2.3 Operationen zur Gültigkeitsprüfung

Operationen, die dazu dienen, einen übergebenen Datensatz auf Gültigkeit zu prüfen, beginnen mit dem Verb

Validate

Beispiel:
Die Operation zur Prüfung der Gültigkeit eines Ticketdatensatzes heißt

`ValidateTicket`

❖ Operations for Validity Checking

Operations used for checking a provided dataset for validity start with verb

Validate

Example:

The operation for checking the validity of a ticket data record is

`ValidateTicket`

4.1.3 Konvention zu Get/Subscribe/Unsubscribe

Zu allen Daten bzw. Datenstrukturen, die man mit Hilfe einer **`Get`**-Operation abrufen kann, existiert auch eine Subscribe- bzw. Unsubscribe-Operation.

Zu allen Daten bzw. Datenstrukturen, auf die man mit Hilfe einer **`Subscribe`**-Operation ein Abonnement einrichten kann, existiert auch eine **`Unsubscribe`**-Operation zur Beendigung des Abonnements und eine **`Get`**-Operation zur einmaligen Abfrage des Werts der Datenstruktur.

Zu jeder **`Unsubscribe`**-Operation auf eine Datenstruktur gehört auch eine **`Subscribe`**-Operation sowie eine **`Get`**-Operation auf dieselbe Datenstruktur.

❖ Conventions for Get/Subscribe/Unsubscribe

A Subscribe and/or Unsubscribe operation exists for all data and/or data structures that can be called up using a **`Get`** operation.

For all data and/or data structures, which can be subscribed using a **`Subscribe`** operation, there is an **`Unsubscribe`** operation for canceling the subscription, and a **`Get`** operation for a single request of a data structure value.

For every **`Unsubscribe`** operation for a data structure, there is a **`Subscribe`** operation as well as a **`Get`** operation for the same data structure.

4.1.4 Unterscheidung zwischen Get und Retrieve

Mit **`Get`**-Operationen greift man immer auf aktuell gültige Daten zu. Der Zugriff erfolgt ohne die Übergabe von Daten oder Datenstrukturen, weil durch den Objektnamen der Operation

festgelegt ist, mit welcher Datenstruktur der Dienst antwortet. Deshalb können hierfür auch Abonnements eingerichtet werden.

`Retrieve`-Operationen beziehen sich nicht auf aktuell gültige Daten, sondern übergeben immer einen Datensatz, von dessen Inhalt die Antwort des Dienstes abhängt.

❖ Differentiation between Get and Retrieve

`Get` operations always access the currently valid data. No data or parameters are provided for the access, as the object name of the operation defines the data structure, the service responds with. For this reason, subscriptions can be set up as well.

`Retrieve` operations do not refer to currently valid data, but always provide a dataset, whose content defines the service response.

4.1.5 Konvention zu Deactivate/Activate

Ein durch eine **`Deactivate`**-Operation deaktiviertes Gerät zeichnet sich dadurch aus, dass es sich im Rahmen der IBIS-IP-Kommunikation, wie ein abgeschaltetes Gerät verhält. D. h. insbesondere, dass keiner der Dienste auf dem Gerät mehr via dem IBIS-IP-Protokoll kommuniziert.

Durch die **`Activate`**-Operation wird ein zuvor deaktiviertes Gerät wieder in Zustand versetzt, den es nach dem Starten des Gerätes hat. Insbesondere sind keine Dienste außer dem *`DeviceManagementService`* nach Ausführung des Activate-Befehls aktiv.

❖ Convention for Deactivate/Activate

A device deactivated via a **`Deactivate`** operation distinguishes itself by its behavior as deactivated device within the scope of IBIS-IP communication. In particular, this means that none of the services on the device communicated via the IBIS-IP protocol.

The **`Activate`** operation switches a previously deactivated device into the state, it had after the device start. In particular, no services except the *`DeviceManagementService`* are active after execution of the Activate command.

4.2 Konventionen für UDP-Dienste

UDP-Dienste werden wie alle fachlichen Dienste, durch das Systemmanagement gestartet. Dabei veröffentlichen UDP-Dienste ihre Daten an eine Multicast-IP-Adresse. Andere Dienste oder Applikationen, die diese Daten abrufen wollen, melden sich an der Multicast-IP-Adresse an und empfangen somit zyklisch neue Daten. Eine Adressierung des Dienstes oder einzelner Operationen dieses Dienstes gibt es nicht.

❖ Conventions for UDP Services

Similar to all functional services, UDP services are started by system management. UDP services publish their data to a multicast IP address. Other services or applications that would like to retrieve this data subscribe to the multicast IP address and receive new data periodically. Service or individual operation addressing does not exist for this service.

4.3 Konventionen für besonderes Dienstverhalten / Conventions for special Service Behavior

4.3.1 Konventionen für Zustände

Dienste in IBIS-IP können in einem der folgenden Zustände sein:

— **notrunning**
In diesem Zustand befinden sich alle Dienste bei Systemstart. Sie bleiben insbesondere in diesem Zustand, sofern sie nicht durch den `SystemManagementService` gestartet werden.

— **running**
In diesen Zustand gelangt ein Dienst erst dann, wenn

  — er selbst sämtliche Daten von anderen Diensten beziehen kann, von denen er abhängt und

  — wenn er daraus gültige Daten ermitteln und bereitstellen kann und

  — wenn er als Dienst per DNS-SD veröffentlicht wurde.

— **defective**
In diesem Zustand gelangt ein Dienst, wenn er sich

  — auch nach Ablauf interner Timeouts nicht mit allen notwendigen Diensten verbinden kann oder

  — wenn keine gültigen Daten mehr bereitgestellt werden können (z. B. durch einen Verbindungsabbruch zu erforderlichen Diensten oder weil die gelieferten Daten nicht verwertbar sind)

— **starting**
In diesem Zustand befindet sich ein Dienst, der gestartet wurde, aber noch nicht den Zustand „running“ erreicht hat. Sollte nach Ablauf eines Timeouts der Zustand „running“ nicht erreicht werden, so wird der Zustand „defective“ gemeldet.

— **standby**
In diesem Zustand gelangt ein Dienst, wenn

  — er selbst sämtliche Daten von anderen Diensten beziehen kann, von denen er abhängt und

  — wenn er daraus gültige Daten ermitteln und bereitstellen kann und

  — wenn er NICHT als Dienst per DNS-SD veröffentlicht wurde.

Der Status der Dienste kann individuell beim jeweiligen `DeviceManagementService` des Gerätes abgefragt werden, auf dem der Dienst läuft. Er wird außerdem zyklisch durch den SystemManagementService abgefragt.

❖ Conventions for States

Services in IBIS-IP can be in one of the following states:

- **notrunning**
  All services are in this state at the time of system start. They remain in this state, unless they are started by the `SystemManagementService`.
- **running**
  A service only enters this state, if it can retrieve all

    - Data, it depends on, from other services, and determine and provide
    - Valid data based on this data,
    - In addition, if it was published as service per DNS-SD.
- **defective**
  A service enters this state, if it cannot connect with
    - All required services after expiration of internal timeouts, or
    - If it cannot provide valid data anymore (e.g. due to a loss of connection to required services or as the delivered data cannot be evaluated)
- **starting**

– A service that was started, but is not yet in the "running" state, is in this state. If the "running" state has not been reached after expiration of a timeout, state "defective" is reported.

- **standby**
  A service enters this state, if it can retrieve all
    - Data, it depends on, from other services, and determine and provide
    - Valid data based on this data,
    - In addition, if it was NOT published as service per DNS-SD.

The service status can be individually requested from the particular `DeviceManagementService` of the device, on which the service is running. It is furthermore periodically requested by SystemManagementService.

4.3.2 Konventionen für Fehlermeldungen

Innerhalb von IBIS-IP gibt es verschiedene Ebenen, auf denen in der Kommunikation Fehler auftreten können und die je nach Ebene unterschiedlich gehandhabt werden.

Kommt es auf der Ebene der Netzwerkkommunikation zu Fehlern, so kommen die Mechanismen der Fehlerbehandlung von TCP bzw. HTTP und UDP zum Einsatz. Wird also beispielsweise eine Anfrage an eine falsche Adresse geschickt, dann wird dies mit einem normalen HTTP-Fehler 404 „File not found“ beantwortet.

Kommt die Netzwerkkommunikation erfolgreich zustande, wird aber für eine Anfrage ein fehlerhaftes XML geschickt, dann wird dies von dem angefragten Dienst mit einer XML-Nachricht beantwortet, in der sich weitergehende Informationen zu der fehlerhaften Anfrage finden. Diese Informationen werden in Form eines frei wählbaren Textes übertragen.

Hat ein Dienst nicht alle Grunddaten, die er zu einer vollständigen Antwort benötigt oder fehlen ihm Dienste, auf die seine eigenen Informationen aufbauen, so ist es in IBIS-IP vorgesehen, dass er trotzdem eine syntaktisch korrekte Antwort gibt, auf die fehlerhalten oder fehlenden Daten jedoch mit einem weiteren Attribut hinweist. Dafür wurden in IBIS-IP die XML-Standarddatentypen durch eigene IBIS-IP-Typen ersetzt, die neben dem eigentlichen Wert auch noch einen Fehlercode enthalten können. Kommt es beispielsweise in einem IBIS-IP-System zu einem Ausfall der GPS-Ortung, weil das Fahrzeug sich in einem Tunnel befindet, so kann der Dienst `NetworkLocationService` trotzdem weiterhin Daten senden, allerdings werden die übermittelten Werte im XML in diesem Fall mit einem ErrorCode „DataEstimated“ versehen, so dass der datenkonsumierende Dienst in diesem Fall die Möglichkeit hat, die gelieferten Informationen zu verwerfen oder sie mit einem entsprechenden Hinweis versehen, weiterzugeben.

❖ Conventions for Error Messages

There are different layers within IBIS-IP, where a communication error can occur, which are differently handled depending on the layer.

If errors occur in the network communication layer, the TCP and/or HTTP and UDP error handling mechanisms are applied. For example, if a request is sent to a wrong address, this will be answered with a normal HTTP error 404 "File not found".

If network communication is successfully established, but a faulty XML is sent for a request, the inquired service responds with a XML message, which contains further information regarding the faulty request. This information is transferred in the form of a freely selectable text.

If a service does not have all basic data required for a complete response, or if services are missing, on which its own information is based, it must be possible to send a response with correct syntax in IBIS-IP. This response must indicate any faulty or missing data with another attribute. For this purpose, the XML standard data types were replaced in IBIS-IP with own IBIS-IP types, which can contain an error code in addition to the actual value. For example, if the GPS locating fails in an IBIS-IP system, as the vehicle is in a tunnel, the `NetworkLocationService` can continue to send data; however, the values transferred in XML are marked with a "DataEstimated" error code. In this case, the data-consuming service can either discard the received information or relay it with a respective note.

4.4 Konventionen für die Datenstrukturierung

Um die Daten innerhalb von IBIS-IP über XML-Schema-Dateien in einer einheitlichen Art und Weise abzubilden, gelten die folgenden Konventionen für die Beschreibung der XML-Strukturen für Operationen:

— Besitzt eine Operation spezifische Frage- und Antwort-Strukturen, so sind diese nach folgendem Schema zu benennen

  — für Anfragen: Dienstname + „.“ + Operationsname + „`RequestStructure`“

  — für Antworten: `Dienstname + „.“ + Operationsname + „ResponseStructure“`

Beispiel:
`<JourneyInformationService.RetrieveAllRoutesPerLineRequestStructure>`

`<JourneyInformationService.RetrieveAllRoutesPerLineResponseStructure>`

— Die Antwort-Strukturen bestehen immer aus einem „choice“ (Auswahlmöglichkeit im XML) zwischen dem eigentlichen Dateninhalt und einer freien Fehlermeldung. Hierbei ist der eigentliche Dateninhalt entsprechend zu benennen und eine Struktur dafür anzulegen, die nach dem Schema

  — `Dienstname + „.“ + Dateninhalt + „Data“`

  zu benennen ist.

Beispiel:
`<JourneyInformationService.AllRoutesData>`

❖ Conventions for Data Structuring

In order to map the data within IBIS-IP via XML schema data in a standardized manner, the following conventions apply to the description of the XML structures for operations:

- If an operation has specific request-response structures, they must be named according to the following schema

  For requests: `Service name + „.“ + Operation name + "RequestStructure"`
  For responses: `Service name + „.“ + Operation name + "ResponseStructure"`

Example:

```
<JourneyInformationService.RetrieveAllRoutesPerLineRequestStructure>

<JourneyInformationService.RetrieveAllRoutesPerLineResponseStructure>
```

- The response structures always consist of a "choice" (selection possibility in XML) between the actual data content and a free error message. Here, the actual data content must be respectively named and a structure created, which must be named according to the following schema

– `Service name + "." + Data content + "Data"`

Example:

```
<JourneyInformationService.AllRoutesData>
```

5 Prinzipien in IBIS-IP

Der Systemstart, das anschließende Starten aller Dienste entsprechend der Konfiguration (vgl. Kapitel 1.2) sowie das Finden der Dienste untereinander erfolgt in IBIS-IP weitgehend automatisch.

Die Findung des Gesamtsystems beruht auf verschiedenen grundlegenden Prinzipien, die in IBIS-IP angewandt werden. Diese Prinzipien werden nachfolgend kurz beschrieben.

❖ **Principles in IBIS-IP**

System start, the subsequent start of all services according to configuration (cf. chapter 1.2), and service discovery among themselves is realized largely automatically in IBIS-IP.

Discovery of the complete system is based on different basic principles, which are applied in IBIS-IP. These principles are briefly described below.

5.1 Grundlagen / Basics

5.1.1 Systemkonfiguration muss vorliegen

In der Systemkonfiguration wird auf rein fachlicher Ebene festgelegt:

— welche Geräte überhaupt am IBIS-IP-System beteiligt sind. Durch diese Information kann eine Fehlermeldung generiert werden, falls eines der Geräte nicht erreichbar ist.

— welcher Dienst auf welchem Gerät in welcher IBIS-IP-Version laufen soll. Mit dieser Information kann der SystemManagementService dafür sorgen, dass auf den Geräten die korrekten Dienste gestartet werden.

Dabei ist zu beachten, dass es nicht erforderlich ist, explizit in der Systemkonfiguration anzugeben, dass ein `DeviceManagementService` (vgl. VDV 301-2-2) auf einem Gerät läuft. Die Angabe eines Gerätes impliziert das Vorhandensein eines `DeviceManagementService` (vgl. Kapitel 1.1)

Beispiel:
In der Systemkonfiguration wird festgelegt, dass der `CustomerInformationService` *(vgl.* VDV 301-2-3*) in der IBIS-IP-Version 1.0 auf dem Gerät OnBoardUnit 1 (vgl.* Kapitel *1.6) laufen soll.*

Für die Konsistenz der Konfiguration ist der Ersteller der Konfiguration verantwortlich. Fehlen Dienste oder Geräte in der Konfiguration, oder werden IBIS-IP-Versionen für Dienste angegeben, mit denen andere Dienstnutzer nicht umgehen können, so wird das System nicht wie geplant funktionieren. Werden entgegen der IBIS-IP Spezifikation Dienste mit gleicher IBIS-IP-Version doppelt (z. B. auf verschiedenen Geräten) gestartet, so kann dies zu Inkonsistenzen führen (vgl. auch Kapitel 5.1.2). Ein solcher Zustand ist daher strikt zu vermeiden.

Die Systemkonfiguration wird vom `SystemDocumentationService` (vgl. VDV 301-2-4) über die Operation ***`GetSystemConfiguration`*** bereitgestellt. Damit können sowohl das Ablageformat als auch der Ablageort der Systemkonfigurationsdaten herstellerspezifisch

implementiert werden, die Inhalte sind jedoch über eine Schnittstelle des `SystemDocumentationService` ansprech- und änderbar.

❖ System Configuration must be available

The system configuration defines the following at a purely functional level:

- The devices participating in the IBIS-IP system. This information can generate an error message, if one of the devices cannot be reached.
- The services that shall run on which device in which IBIS-IP version. Using this information, SystemManagementService can ensure that the correct services are started in the devices.

Please note that it must not explicitly be configured in the system configuration that a `DeviceManagementService` (cf. VDV 301-2-2) is running on a device. The declaration of a device implies the presence of a `DeviceManagementService` (cf. chapter 1.1)

Example:

The system configuration specifies that the `CustomerInformationService` *(cf.* VDV 301-2-3*) should run in IBIS-IP version 1.0 on the OnBoardUnit 1 device (cf. chapter 1.6).*

The author of the configuration is responsible for the consistency of the configuration. If services or devices are missing in the configuration, or if IBIS-IP versions are declared for the services, no other service user can handle, the system will not function as planned. If contrary to the IBIS-IP specification, services with the same IBIS-IP version are started twice (e.g. on different devices), this can result in inconsistencies (also cf. chapter 5.1.2). Thus, such a state must be strictly avoided.

The system configuration is provided by the `SystemDocumentationService` (cf. VDV 301-2-4) via the ***`GetSystemConfiguration`*** operation. In this context, the format as well as the location of the system configuration data can be implemented manufacturer-specifically. However, the contents can only be addressed and changed via an interface of the `SystemDocumentationService`.

5.1.2 Nur ein Dienst pro Fachlichkeit

In IBIS-IP darf nur genau ein Dienst für eine fachliche Aufgabe im System gestartet sein. Damit werden Zustände vermieden, in denen es durch mehrfach vorhandene Dienste zu Mehrdeutigkeiten kommt.

Beispiel:
Würden in einem System zwei `CustomerInformationServices` *laufen, so wäre es (aufgrund von* Kapitel *5.1.3) nicht klar, welcher Fahrgast-Anzeiger seine Informationen von welchem der beiden* `CustomerInformationServices` *holt. Insbesondere kann es zu Zuständen kommen, in denen zwei Anzeiger zumindest zwischenzeitlich verschiedene Informationen anzeigen.*

Die bedeutet nicht, dass es nicht zwei Dienste gleichen Typs in einem IBIS-IP-System geben darf. Sie müssen aber sowohl fachlich verschiedene Aufgaben im IBIS-IP-System übernehmen als auch auf verschiedenen technischen Wegen adressierbar sein.

Beispiel 1:

In einem IBIS-IP-System mit 5 Geräten, gibt es auch fünf `DeviceManagementServices` Diese erfüllen jedoch fachlich verschiedene Aufgaben (sie managen verschiedene Geräte) und sie sind technisch auf verschiedenen Wegen adressierbar.

Beispiel 2:
Wenn in einem Fahrzeug mit drei Türen drei Fahrgastzählgeräte angebracht sind, so kann jedes dieser Geräte einen Dienst zur Abholung der Fahrgastzahlen bereitstellen. Die Geräte erfüllen fachlich verschiedene Aufgaben (sie zählen Ein-/Aussteiger an verschiedenen Türen) und sind technisch auf verschiedenen Wegen adressierbar.

Beispiel 3:
Wenn in einem IBIS-IP-System ein älterer Fahrgastanzeiger ausschließlich mit Daten des `CustomerInformationService` der IBIS-IP-Version 1.0 umgehen kann, man aber gleichzeitig auf einem moderneren Fahrgastanzeiger Features des `CustomerInformationService` der IBIS-IP-Version 2.0 nutzen will, so ist es möglich, zwei `CustomerInformationServices` im IBIS-IP-System zu betreiben. Die Dienste erfüllen fachlich verschiedene Aufgaben und sind technisch auf verschiedenen Wegen adressierbar. Die Adressierung verschiedener Versionen eines Dienstes auf einem Gerät erfolgt durch Verwendung verschiedener Ports oder verschiedener Pfade für die verschiedenen Versionen des Dienstes.

Die Informationen über die in einem IBIS-IP vorhandenen Geräte bzw. alle benötigten Dienste in den benötigten IBIS-IP-Versionen werden vom `SystemDocumentationService` bereitgestellt.

Eine Sonderrolle spielt der `SystemManagementServivce`, der tatsächlich nur genau ein einziges Mal im gesamten IBIS-IP-System vorkommen darf (also auch in nur genau einer IBIS-IP-Version). Auf diese Weise wird sichergestellt, dass nicht durch den Parallelbetrieb zweier `SystemManagementServices` zwei parallele IBIS-IP-Systeme verwaltet werden.

❖ One Service only per Functionality

One service only may be started per functionality in the system in IBIS-IP. This avoids states, in which ambiguity occurs due to services existing multiple times.

Example:

If two `CustomerInformationServices` would run in a system, it would not be clear (due to chapter 5.1.3), which passenger display gets its information from which of the two `CustomerInformationServices`. In particular, states can occur, where two displays show - at least temporarily - different information.

This does not mean that there cannot be two services of the same type in an IBIS-IP system. However, they must assume different functional tasks in the IBIS-IP system, and must be addressable via different functional paths.

Example 1:

In an IBIS-IP system with 5 devices, five `DeviceManagementServices` exist. However, they fulfill functionally different tasks (they manage different devices), and can be technically addressed via different paths.

Example 2:

If three passenger counting devices are installed in a vehicle with three doors, each of these devices can provide a service for retrieving the passenger numbers. The devices meet functionally different tasks (they count passengers entering/exiting through the different doors), and can be technically addressed via different paths.

Example 3:

If an older passenger display in an IBIS-IP system can only handle data of the `CustomerInformationService` of IBIS-IP version 1.0, but features of `CustomerInformationService` of IBIS-IP version 2.0 should be used at the same time on a modern passenger display, two `CustomerInformationServices` can be operated in the IBIS-IP system. The services fulfill functionally different tasks can be technically addressed via different paths. Different versions of a service are addressed on a system by using different ports or paths for the different versions of the service.

Information about the devices present in an IBIS-IP and all required services in the required IBIS-IP versions are provided by the `SystemDocumentationService`.

The `SystemManagementServivce` plays a special role. It must only exist once within the complete IBIS-IP system (hence, also in exactly one IBIS-IP version only). This way, it is ensured that the parallel operation of two `SystemManagementServices` does not result in two parallel IBIS-IP systems.

5.1.3 Kenntnis der erforderlichen Dienste

In der Systemkonfiguration wird NICHT festgelegt, welche Applikation bzw. Dienst sich mit welchem Dienst verbinden soll. Die Information, von welchen anderen Diensten mit welcher IBIS-IP-Version eine Applikation oder ein Dienst abhängig ist, ist immer Teil der Implementierung der Applikation bzw. des Dienstes selbst.

Der Dienst bzw. die Applikation verbindet sich immer mit dem Dienst, der als erstes mit passendem Dienstnamen und passender IBIS-IP-Version via DNS-SD gefunden wird (vgl. Kapitel 3.1und 5.1.4.2). Deshalb ist es wichtig, durch die Systemkonfiguration sicherzustellen, dass es nur genau einen Dienst pro Fachlichkeit gibt.

Wenn ein Dienst bzw. eine Applikation Daten verschiedener IBIS-IP-Versionen eines anbietenden Dienstes handhaben kann, so ist auf Seiten des nutzenden Dienstes bzw. der nutzenden Applikation festzulegen, welche Version zu verwenden bzw. zu bevorzugen ist.

Beispiel:
Die Applikation auf einem Fahrgast-Anzeiger ist so implementiert, dass sie immer nach einem Dienst `CustomerInformationService` *in der IBIS-Version 2.0 sucht und von diesem Dienst Daten abruft und anzeigt, sobald der Dienst für die Applikation auf dem Fahrgast-Anzeiger verfügbar ist.*

❖ Knowledge of the required Services

The system configuration does NOT define, which application and/or service should be connected with which service. The information, from which other services with which IBIS-IP version an application or service depends, is always included in the implementation of the application and/or service itself.

The service and/or application always connects with the service, which is found first with matching service name and matching IBIS-IP version via DNS-SD (cf. chapters 3.1 and 5.1.4.2). For this reason, it is important that the system configuration ensure that only one service exists per functionality.

If a service and/or application can handle the data of different IBIS-IP version of an offering service, the service and/or application using the data must define which version should be used and/or preferred.

Example:

The application on a passenger display must be implemented such that it always searches for a `CustomerInformationService` *in IBIS version 2.0 and retrieves and displays data from this service, as soon as the service is available to the application on the passenger display.*

5.1.4 Identifikation von Diensten und Geräten in IBIS-IP

Nachfolgend wird beschrieben, wie Geräte und Dienste in IBIS-IP identifiziert werden.

Dabei wird unterschieden zwischen einer fachlichen Identifikation und einer technischen Identifikation.

Fachliche Identifikation

Die fachliche Identifikation dient der Beschreibung und Konfiguration des Systems. Hierbei liegt der Fokus auf einer verständlichen Darstellung des Systems. Geräte und Dienste können mit Hilfe der fachlichen Identifikation konfiguriert werden, ohne dass dabei die spezifische technische Umsetzung, also IP-Adressen, Ports oder Pfade der Dienste, bekannt sein müssen.

Fachliche Identifikationsdaten werden insbesondere bei Systemstart zwischen dem `SystemManagementService` und dem `SystemDocumentationService` ausgetauscht, um die System-Konfiguration des IBIS-IP-Systems zu ermitteln und die korrekten Dienste zu starten. Fachliche Identifikationsdaten sind also für den System- bzw. Dienststart relevant.

Technische Identifikation

Die technische Identifikation hingegen dient der IP-basierten Kommunikation mit den Diensten in IBIS-IP. Technische Identifikationsdaten lassen sich per DNS-SD ermitteln und sind für die Dienstnutzung relevant.

❖ Service and Device Identification in IBIS-IP

In the following it is described, how devices and services are identified in IBIS-IP.

In this context, it is differentiated between a functional and technical identification.

Functional Identification

The functional identification is used for system description and configuration. The focus is on a comprehensible representation of the system. Devices and services can be configured using the functional identification. For this purpose, no information must be known about the specific technical implementation, i.e. IP addresses, ports or paths of the services.

Functional identification data is in particular exchanged at the time of system start between the `SystemManagementService` and the `SystemDocumentationService` in order to determine the system configuration of the IBIS-IP system and to start the correct services. Thus, functional identification data is relevant to the system and/or service start.

Technical Identification

Technical identification is used for IP-based communication with the services in IBIS-IP. Technical identification data can be determined per DNS-SD and is relevant for service use.

5.1.4.1 Geräte

Die Zuordnung von fachlicher zu technischer Identifikation eines Gerätes erfolgt durch den `SystemManagementService`. Der `SystemManagementService` ist der einzige Dienst, der überhaupt Geräte überwacht und managt (vgl. Kapitel 7.3) bzw. eine Zuordnung zwischen fachlicher und technischer Identifikation vornimmt.

Alle anderen Dienste und Applikationen kommunizieren nur mit Diensten, die sich per DNS-SD anhand ihrer technischen Identifikation adressieren lassen, d. h., insbesondere ohne Berücksichtigung der Geräte, auf denen die Dienste laufen oder sonstiger fachlicher Identifikationsdaten.

❖ Devices

`SystemManagementService` assigns the functional identification of a device to the technical identification. `SystemManagementService` is the only service that monitors and manages devices (cf. chapter 7.3) and/or that assigns functional identifications to technical identifications.

All other services and applications only communicate with the services that can be addressed per DBS-SD based on their technical identification. This means in particular, without considering the devices, on which the services are running, or other functional identification data.

5.1.4.1.1 Fachliche Identifikation

Ein Gerät wird in IBIS-IP systemweit eindeutig durch

— einen Gerätetyp und

— eine Geräte-ID

identifiziert.

Mögliche Gerätetypen finden sich in Abschnitt 1.4 aufgelistet.

Eine Geräte-ID ist eine eindeutige Identifikationsnummer, die einen Rückschluss auf den Einbauort des Gerätes erlaubt. Die Kombination aus Gerätetyp und Geräte-ID erlaubt eine systemweit eindeutige fachliche Identifikation.

Beispiel:
Ein Entwerter-Gerät (Validator), der sich an der Einbauposition 3 befindet, kann fachlich als Entwerter 3 (Validator 3)bezeichnet werden.

❖ Functional Identification

A device is clearly identified across the IBIS-IP system using a

- Device type, and a
- Device ID.

Possible device types are listed in section 1.4.

A device ID is a unique identification number that allows a conclusion regarding the installation location of the device. The combination of device type and device ID allows a clear functional identification across the system.

Example:

A validator located at installation position 3 can functionally be referred to as validator 3.

5.1.4.1.2 Technische Identifikation

Technisch kann ein Gerät innerhalb des Netzwerks über

— seine systemweit eindeutige IP-Adresse oder

— seinen systemweit eindeutigen DNS-Namen

identifiziert werden.

❖ Technical Identification

A device can be technically identified within the network

- Using its system-wide unique IP address or
- Its system-wide unique DNS name.

5.1.4.2 Dienste / Services

5.1.4.2.1 Fachliche Identifikation

Ein Dienst wird systemweit fachlich eindeutig identifiziert durch

— Dienstname

— IBIS-IP-Version

— Gerätetyp und

— Geräte-ID.

Das Konzept erlaubt es insbesondere, dass mehrere Versionen eines IBIS-IP-Dienstes auf einem Gerät vorhanden sind und parallel in Betrieb sind. (Ausnahme SystemManagementService, vgl. Kapitel 7.3)

Beispiel:
Der `CustomerInformationService` *mit IBIS-IP-Version 1.0 auf der Geräteklasse OnBoardUnit mit Geräte ID 1 ist systemweit eindeutig identifiziert.*

❖ Functional Identification

A service is functionally uniquely identified across the entire system by

- Service name
- IBIS-IP version
- Device type, and
- Device ID.

The concept in particular allows that multiple versions of an IBIS-IP service are present on a device and operate in parallel. (Exception SystemManagementService, cf. chapter 7.3)

Example:

The `CustomerInformationService` *with IBIS-IP version 1.0 is clearly identified across the entire service in device class OnBoardUnit with device ID 1.*

5.1.4.2.2 Technische Identifikation

Ein Dienst wird systemweit technisch eindeutig identifiziert durch

— die systemweit eindeutige IP-Adresse oder den systemweit eindeutigen DNS-Namen des Gerätes auf dem der Dienst läuft

— den Dienstnamen,

— den Ports und

— (*optional*) des Pfads, unter dem der Dienst ansprechbar ist.

Verschiedene Versionen eines Dienstes müssen von einem Gerät dabei unter verschiedenen Ports oder verschiedenen Pfaden veröffentlicht werden.

Der Protokolltyp (_ibisip_udp bzw. _ibisip_http, vgl. Kapitel 3.4 und 3.5) wird nicht zur Identifikation herangezogen, da der Protokolltyp sich direkt aus dem Dienst ergibt und ein Dienst nur von einem der beiden Typen sein kann.

Beispiel 1:
Der Dienst `CustomerInformationService` *aus VDV 301-2-3 bietet seine Operationen der IBIS-IP-Version 1.0 unter dem Port 34567 und dem Pfad „/1.0/“ an. Operationen der IBIS-IP-Version 2.0 werden dagegen unter dem Port 34567 und dem Pfad „/2.0/“ angeboten.*

Dabei ist der `CustomerInformationService` *immer ein _ibisip_http-Dienst.*

Beispiel 2:
Der `CustomerInformationService` *aus VDV 301-2-3 wird technisch systemweit eindeutig identifiziert durch*

`<192.168.47.11:34567/1.0/CustomerInformationService>` *oder*

`<vdv-test-dev:34567/1.0/CustomerInformationService>`

wobei `192.168.47.11` *die IP-Adresse,* `vdv-test-dev` *der zugehörige DNS-Name,* `34567` *der Port,* `1.0` *der Pfad und* `CustomerInformationService` *der Dienstname ist.*

❖ Technical Identification

A service is technically uniquely identified across the entire system by

- The IP address unique across the system or the DNS name unique across the system of the device the service runs on,
- Service name,
- Ports, and
- (optional) Path, under which the service can be addressed.

Different versions of a service must be published by a device under different ports or different paths.

The protocol type (_ibisip_udp and/or _ibisip_http, cf. chapters 3.4 and 3.5) is not used for identification, as the protocol type directly results from the service and a service can only be of one of these two types.

Example 1:

Service `CustomerInformationService` *from VDV 301-2-3 offers its operations of IBIS-IP version 1.0 under port 34567 and path "/1.0/". Operations of IBIS-IP version 2.0 are offered under port 34567 and path "/2.0/".*

`CustomerInformationService` *is always a _ibisip_http service.*

Example 2:

`CustomerInformationService` *from VDV 301-2-3 is technically uniquely identified across the entire system by*

`<192.168.47.11:34567/1.0/CustomerInformationService>` *or*

`<vdv-test-dev:34567/1.0/CustomerInformationService>`

where `192.168.47.11` *is the IP address,* `vdv-test-dev` *the corresponding DNS name,* `34567` *the port,* `1.0` *the path, and* `CustomerInformationService` *the service name.*

5.2 Das Konzept des Systemstarts

Nachfolgend ist der Systemstart für ein System beschrieben, bei dem fachlich gleiche Dienste im Bedarfsfall von verschiedenen Geräten bereitgestellt werden können. Dazu ist eine zentrale Steuerung und Konfiguration erforderlich.

> Best-Practice-Hinweis:
> In den meisten IBIS-IP-Projekten ist es ausreichend, wenn die Dienst-zu-Gerät-Zuordnung statisch ist, sich also während des Betriebs nie ändert. Deshalb ist es i.d.R. so, dass beim Geräte-Start immer die gleichen Dienste gestartet werden.
> Auf die Implementierung eines `SystemManagementService` und eines `SystemDocumentationService` kann dann verzichtet werden.

Der Start eines IBIS-IP-Systems ist ein zweistufiger Prozess. Bei beiden Stufen werden im Wesentlichen folgende Teilschritte durchlaufen:

— Start der Dienste

— Veröffentlichung der Dienste

— Verbindungsaufbau mit den veröffentlichten Diensten und Ausführung fachlicher Aufgaben

In der ersten Stufe erfolgt der Teilschritt „Start der Dienste“ automatisch beim Start der Geräte. Es werden dabei nur die systemrelevanten Dienste der Fahrzeugbetriebsfunktionen“ (`DeviceManagementService`, `SystemManagementService` und `SystemDocumentationService`) gestartet.

In der zweiten Stufe wird der Teilschritt „Start der Dienste“ durch den dann laufenden `SystemManagementService` gesteuert.

❖ The System Start Concept

The start of an IBIS-IP system is a two-stage process. The following sub-steps are mainly executed in both stages:

- Start services
- Publish services
- Connection establishment with the published services and execution of functional tasks

In the first stage, sub-step "Start services" is automatically executed, when the device is started. In this context, the system-relevant services of the vehicle operation functionalities (`DeviceManagementService`, `SystemManagementService` and `SystemDocumentationService`) are started only.

In the second stage, sub-step "Start services" is controlled by the running `SystemManagementService`.

Best Practice Note:

In most IBIS-IP projects, it is sufficient if the service-to-device assignment is static, i. e. it never changes during operation. For this reason, it is usually the case that the same services are always started when the device is started.

The implementation of a `SystemManagementService` and a `SystemDocumentationService` can then be dispensed with.

5.2.1 Stufe 1: Start systemrelevanter Fahrzeugbetriebsfunktionen / Stage 1: Start of system-relevant Vehicle Operation functionalities

5.2.1.1 Start der Dienste der Fahrzeugbetriebsfunktionen

Beim Start der Geräte eines IBIS-IP-Systems startet auf jedem der Geräte zuerst der Dienst `DeviceManagementService` (vgl. VDV 301-2-2).

Für jedes dieser Geräte ist festgelegt, ob weitere Dienste der Fahrzeugbetriebsfunktionen (d. h. `SystemDocumentationService` und/oder `SystemManagementService`) auf dem Gerät automatisch gestartet werden sollen. Die Festlegung, dass und ggf. welche Dienste automatisch gestartet werden, erfolgt durch eine geräte- bzw. herstellerspezifische Konfiguration, die nicht im Rahmen von IBIS-IP standardisiert, jedoch über eine Schnittstelle des `DeviceManagementServices` abrufbar und änderbar ist.

Die Festlegung muss so erfolgen, dass nach Start aller Geräte und Durchführung des Starts aller automatischen zu startenden Dienste der Fahrzeugbetriebsfunktionen folgende Situation eintritt:

— Auf jedem Gerät des IBIS-IP-Systems läuft genau ein `DeviceManagementService`

— Auf genau einem der Geräte des IBIS-IP-Systems läuft genau ein `SystemDocumentationService`. Es gibt keinen weiteren `SystemDocumentationService` im IBIS-IP-System (vgl. auch VDV 301-2-4).

— Auf genau einem der Geräte des IBIS-IP-Systems läuft genau ein `SystemManagementService`. Es gibt keinen weiteren `SystemManagementService` im IBIS-IP-System. (vgl. auch Kapitel 7.3).

Beispiel:
In einem System aus einem Bordrechner und mehreren Anzeigern starten durch den Start der Geräte zunächst alle `DeviceManagemetnServices`. Auf dem Bordrechner werden durch den `DeviceManagementService` zusätzlich die Dienste `SystemDocumentationServcie` und `SystemManagementService` gestartet. Auf allen Anzeigern bleiben dagegen die `DeviceManagementServices` die einzigen vorerst laufenden Dienste.

(Anmerkung: In Abbildung 5 im Kapitel 5.3.2.3 wird der etwas komplexere Fall beschrieben, bei dem der `SystemDocumentationService` und der `SystemManagementService` auf verschiedenen Geräten betrieben werden.)

❖ Start of the Services of the Vehicle Operation Functionalities

When the devices of an IBIS-IP system are started, service `DeviceManagementService` starts first on every device (cf. chapter VDV 301-2-2).

For every one of these devices it is defined, whether further services of the vehicle operation functionalities (i.e. `SystemDocumentationService` and/or `SystemManagementService`) should be automatically started on the device. The definition that and possibly which services are automatically started, is provided by a device- and/or manufacturer- specific configuration, which is not standardized within IBIS-IP, but which can be retrieved and changed via an interface of the `DeviceManagementServices`.

The definition must ensure that the following situation occurs after all devices are started and all services of the vehicle operation functionalities to be automatically started are started:

- Exactly one `DeviceManagementService` runs on every device of the IBIS-IP system
- Exactly one `SystemDocumentationService` runs on exactly one of the devices of the IBIS-IP system. No further `SystemDocumentationService` exists in the IBIS-IP system (also cf. chapter VDV 301-2-4).
- Exactly one `SystemManagementService` runs on exactly one of the devices of the IBIS-IP system. No further `SystemManagementService` exists in the IBIS-IP system (also cf. chapter 7.3).

Example:

In a system of one on-board computer and several displays, by the devices start all `DeviceManagementServices` *are started. The* `DeviceManagementService` *starts the* `SystemDocumentationService` *and* `SystemManagementService` *in addition on the on-board computer. However, for the time being, the* `DeviceManagementServices` *remain the only services running on the displays.*

(Remark: In figure 6, chapter 5.3.2.3, the slightly more complex case is described, where `SystemDocumentationService` and `SystemManagementService` are operated on different devices.)

5.2.1.2 Veröffentlichung der Dienste der Fahrzeugbetriebsfunktionen

Alle Dienste in IBIS-IP nutzen den in Kapitel 3 beschriebenen Mechanismus der Veröffentlichung per DNS-SD.

Durch die Veröffentlichung werden folgende Informationen zu diesem Dienst systemweit bekannt gemacht:

— Dienstname

— IBIS-IP-Version des Dienstes

— Protokoll, über das der Dienst angesprochen werden kann (bei Diensten der Fahrzeugbetriebsfunktionen immer `_ibisip_http`

— IP-Adresse bzw. DNS-Name des Gerätes, auf dem der Dienst läuft

— Port, unter dem der Dienst zu erreichen ist

— Ggf. Pfad, unter dem der Dienst zu erreichen ist

❖ Publication of the Services of the Vehicle Operation Functionalities

All services in IBIS-IP use the publication mechanism via DNS-SD described in chapter 3.

With the publication, the following information regarding this service is announced across the system:

- Service name
- IBIS-IP version of the service
- Protocol, using which the service can be addressed (in the case of vehicle operation functionalities services, always `_ibisip_http`
- IP address and/or DNS name of the device, on which the service is running
- Port, under which the service must be reached
- Path, under which the service must be reached (as needed)

5.2.1.3 Verbindungsaufbau mit den veröffentlichten Diensten der Fahrzeugbetriebsfunktionen und Ausführung fachlicher Aufgaben

Nach der Veröffentlichung der Dienste können die Dienste insbesondere des *SystemManagementService* angesprochen werden. Dabei werden die nachfolgend beschriebenen drei Teilschritte durchlaufen:

❖ Establish Connection with the published Services of the Vehicle Operation Functionalities and Execution of functional Tasks

After the services are published, in particular the services of the *SystemManagementService* can be addressed. In this context, the three sub-steps described below are executed:

5.2.1.3.1 Abfrage der Systemkonfiguration

Der *SystemManagementService* erfragt zunächst beim *SystemDocumentationService* die Konfiguration des IBIS-IP-Systems und erhält eine fachliche Beschreibung des Systems, in der festgelegt ist, auf welchem Gerät (identifiziert durch Gerätetyp und Geräte-ID) welcher Dienst (identifiziert durch Dienstname und IBIS-IP-Version) laufen soll (vgl. auch Kapitel 5.1.1).

❖ System Configuration Request

The *SystemManagementService* first requests the configuration of the IBIS-IP system from the *SystemDocumentationService* and receives a functional description of the system. This description defines, which device (identified via device type and device ID) should run which service (identified via service name and IBIS-IP version) (also cf. chapter 5.1.1).

5.2.1.3.2 Abfrage der DeviceManagementServices

Der SystemManagementService verbindet sich anschließend mit allen verfügbaren *DeviceManagementServices*. Zu diesem Zeitpunkt ist dem *SystemManagementService* noch nicht bekannt, mit welchem *DeviceManagementService* auf welchem Gerät es kommuniziert. Deren technische Verbindungsdaten sind ihm aber durch die Veröffentlichung via DNS-SD bekannt.

Der *SystemManagementService* erfragt bei den *DeviceManagementServices* der Geräte insbesondere

— von welchem Gerätetyp das zugehörige Gerät ist und

— welche Geräte-ID es hat und

— welche Dienste auf dem Gerät verfügbar sind.

❖ DeviceManagementServices Request

Next, the SystemManagementService connects with all available *DeviceManagementServices*. Now, the *SystemManagementService* does not know yet, which *DeviceManagementService* is used on which device for communication. However, it knows their technical connection data based on the publication via DNS-SD.

The *SystemManagementService* requests from the *DeviceManagementServices* of the device in particular the following

- Device type of the corresponding device,
- Device ID, and
- Services available on the device.

5.2.1.3.3 Zuordnung technischer und fachlicher Verbindungsdaten

Durch Kombination der technischen Verbindungsdaten aus der Veröffentlichung mit den fachlichen Informationen über Gerätetyp und Geräte-ID ist es dem *SystemManagementService* möglich, beide Daten gegenseitig korrekt zuzuordnen.

❖ Assignment of technical and functional Connection Data

The combination of the technical connection data from the publication of the functional information about device type and device ID allows the *SystemManagementService* to assign both sets of data with one another.

5.2.1.3.4 Startbefehle an die DeviceManagementServices

Mit dieser Zuordnung kann der *SystemManagementService* entsprechend den fachlichen Konfigurationsdaten vom *SystemDocumentationService* sich auch mit den *DeviceManagementServices* der einzelnen Geräte technisch korrekt verbinden, um auf dem betreffenden Gerät weitere Dienste zu starten.

Sollte der *SystemManagementService* feststellen, dass Dienste auf Geräten nicht verfügbar sind, auf denen sie aber laut Konfiguration laufen sollen, so erkennt dies der *SystemManagementService* und speichert den Fehlerzustand.

Sollte der *SystemManagementService* feststellen, dass Geräte nicht verfügbar sind, die laut Konfiguration vorhanden sein sollen, so erkennt dies der *SystemManagementService* und speichert den Fehlerzustand.

❖ Start Commands to the DeviceManagementServices

Using this assignment, the *SystemManagementService* can technically correctly connect with the *DeviceManagementServices* of the individual devices according to the functional configuration data of the *SystemDocumentationService* in order to start further service on the respective devices.

If the *SystemManagementService* determines that services are not available on devices, where they should be running according to the configuration, the respective error state is stored by the *SystemManagementService*.

If the *SystemManagementService* determines that devices are not available, which should be available according to the configuration, the respective error state is stored by the *SystemManagementService*.

5.2.2 Stufe 2: Start der fachlichen Dienste / Stage 2: Start functional Services

5.2.2.1 Start weiterer Dienste auf den Geräten

Nachdem der *SystemManagementService* sowohl die Systemkonfiguration kennt als auch alle *DeviceManagementServices* der beteiligten Geräte zuordnen und ansprechen kann, ist der *SystemManagementService* in der Lage, weitere Dienste des IBIS-IP-Systems auf den Geräten zu starten.

Dazu ruft der *SystemManagementService* Operation **StartService** bei den *DeviceManagementServices* der beteiligten Geräte auf. Auf diese Weise werden nacheinander alle weiteren Dienste gestartet, die in der Konfiguration angegeben sind.

❖ Start further Services on the Devices

Once the *SystemManagementService* knows the system configuration and can assign and address all *DeviceManagementServices* of the involved devices, the *SystemManagementService* will be able to start further services of the IBIS-IP system on the devices.

For this purpose, the *SystemManagementService* calls up the **StartService** operation for the *DeviceManagementServices* of the involved devices. This way, all further services declared in the configuration are sequentially started.

5.2.2.2 Veröffentlichung weiterer Dienste

Alle Dienste in IBIS-IP nutzen den in Kapitel 3 beschriebenen Mechanismus der Veröffentlichung per DNS-SD.

Durch die Veröffentlichung werden folgende Informationen zu diesen Diensten systemweit bekannt gemacht:

Bei HTTP-Diensten

— Dienstname,

— IBIS-IP-Version des Dienstes,

— Protokoll, über das der Dienst angesprochen werden kann,

— IP-Adresse bzw. DNS-Name des Gerätes, auf dem der Dienst läuft,

— Port, unter dem der Dienst zu erreichen ist,

— Ggf. Pfad, unter dem der Dienst zu erreichen ist.

Bei UDP-Diensten

— Dienstname,

— IBIS-IP-Version des Dienstes,

— Protokoll, über das der Dienst angesprochen werden kann,

— IP-Adresse der Multicast-Gruppe, über die die Informationen des Dienstes verbreitet werden.

❖ Publish further Services

All services in IBIS-IP use the publication mechanism via DNS-SD described in chapter 3.

With the publication, the following information regarding these services is announced across the system:

For HTTP services

- Service name,
- IBIS-IP version of the service,
- Protocol, over which the service can be addressed,
- IP address and/or DNS name of the device, where the service is running,
- Port, under which the service must be reached,
- Path, under which the service must be reached (as needed),

For UDP services

- Service name,
- IBIS-IP version of the service,
- Protocol, over which the service can be addressed,
- IP address of the multicast group, over which the information of the service is distributed.

5.2.2.3 Verbindungsaufbau mit den veröffentlichten Diensten und Ausführung fachlicher Aufgaben

Nach der Veröffentlichung aller benötigten bzw. konfigurierten Dienste werden die informationskonsumierenden Dienste und Applikationen entweder die entsprechenden Abfragen oder die gewünschten Daten-Abonnements einrichten.

Zur Sicherstellung der Funktionsfähigkeit von Datenabonnements (da es Datensendungen nur bei Änderung der Informationen gibt, wissen die Datenkonsumenten in aller Regel nicht, ob ein Dienst noch tätig ist) wird empfohlen, dass die Dienste und Applikationen, die ein Datenabonnement bei einem Dienst einrichten, sich zusätzlich über den aktuellen Systemzustand informieren lassen.

Beispiel:
Die Applikation auf einem Fahrgast-Anzeiger ist so implementiert, dass sie immer nach einem Dienst `CustomerInformationService` *in der IBIS-Version 2.0 sucht und von diesem Dienst Daten abruft und anzeigt, sobald der Dienst für die Applikation auf dem Fahrgast-Anzeiger verfügbar ist. Es ist insbesondere nicht erforderlich, durch eine manuelle Konfiguration diese Festlegung zu treffen*

❖ Establish Connection with the published Services and Execution of functional Tasks

After publishing all required and/or configured services, the information-consuming services and applications will set up the respective requests or the desired subscriptions to data.

In order to ensure the proper functioning of subscriptions to data (as data is only sent in the case of information changes, the data consumers normally do not know, whether a service is still functioning), it is recommended that services and applications that set up subscriptions to a service, are also informed about the current system state.

Example:

The application on a passenger display must be implemented such that it always searches for a `CustomerInformationService` *in IBIS version 2.0 and retrieves and displays data from this service, as soon as the service is available to the application on the passenger display. In particular, it is not necessary to define this via a manual configuration*

5.3 Beispiel eines Systemstarts in IBIS-IP

Nachfolgend ist der Systemstart eines IBIS-IP-Systems für eine Beispielkonfiguration beschrieben, an der zwei Geräte, Gerät A und Gerät B, beteiligt sind.

Auf Gerät A laufen folgende Dienste

— `DeviceManagementService`

— `SystemManagementService`

— `Dienst X`

Auf Gerät B laufen folgende Dienste:

— `DeviceManagementService`

— `SystemDocumentationService`

— *`Dienst Y`*, bei dem man ein Abo einrichten kann.

Der *`SystemManagementService`* fragt außerdem regelmäßig per Statusabfrage bei beiden *`DeviceManagementServices`* den Status der Dienste auf den Geräten sowie den Status des Gerätes ab (vgl. auch 7.3 und VDV 301-2-2).

Auch systemrelevante Dienste der Fahrzeugbetriebsfunktionen werden im Beispiel vom *`DeviceManagementService`* des Gerätes gestartet. Dies kann aber auch auf anderem Weg (z. B. durch ein Skript, eine Applikation oder eine sonstige Autostart-Funktion) geschehen.

❖ Example of a System Start in IBIS-IP

The system start of an IBIS-IP system for an example configuration is described, which involves two devices, device A and device B.

The following services are running on device A

- *`DeviceManagementService`*

- *SystemManagementService*
- *Service X*

The following services are running on device B

- *DeviceManagementService*
- *SystemDocumentationService*
- *Service Y*, where a subscription can be set up.

The *SystemManagementService* also regularly issues a status request to both *DeviceManagementServices* with respect to the service status on the devices and the device status (cf. chapter 7.3 and VDV 301-2-2).

In the example, system-relevant services of the vehicle operation functionalities are also started by the *DeviceManagementService* of the device. However, this is also possible differently (e.g. by script, application, or other autostart functions).

5.3.1 Stufe 1: Start systemrelevanter Fahrzeugbetriebsfunktionen / Stage 1: Start of system-relevant Vehicle Operation Functionalities

5.3.1.1 Start der Dienste der Fahrzeugbetriebsfunktionen

Die *DeviceManagementServices* beider Geräte starten automatisch durch den Gerätestart.

Auf Gerät A sorgt dann die gerätespezifische Konfiguration dafür, dass durch den *DeviceManagementService* des Gerätes A auch der Dienst *SystemManagementService* gestartet wird.

Auf Gerät B sorgt dann die gerätespezifische Konfiguration dafür, dass durch den *DeviceManagementService* des Gerätes B auch der Dienst *SystemDocumentationService* gestartet wird.

❖ Start of the Services of the Vehicle Operation Functionalities

The *DeviceManagementServices* of both devices are automatically started via the device start.

On device A, the device-specific configuration ensures that the *DeviceManagementService* of device A also starts the *SystemManagementService*.

On device B, the device-specific configuration ensures that the *DeviceManagementService* of device B also starts the *SystemDocumentationService*.

5.3.1.2 Veröffentlichung der Dienste der Fahrzeugbetriebsfunktionen

Diese Stufe wird zeitlich versetzt durchlaufen. Jeder Dienst veröffentlicht sich, sobald er betriebsbereit ist. Die *DeviceManagementServices* veröffentlichen sich zeitlich also vor den anderen Diensten der Fahrzeugbetriebsfunktionen.

❖ Publication of the Services of the Vehicle Operation Functionalities

This stage is executed with a time offset. Every service publishes itself, as soon as it is operational. Thus, the *DeviceManagementServices* are published earlier than the other services of the vehicle operation functionalities.

5.3.1.3 Verbindungsaufbau mit veröffentlichten Diensten der Fahrzeugbetriebsfunktionen und Ausführung fachlicher Aufgaben

Der *SystemManagementService* wird per DNS-SD informiert, wenn ein neuer Dienst zur Dienstliste von DNS-SD hinzukommt. Sobald ein *SystemDocumentationService* in dieser Liste auftaucht, verbindet sich der *SystemManagementService* mit diesem Dienst und fragt dort die System-Konfiguration (in der Abbildung 5, SDS/GetSystemConfiguration) ab.

Sobald ein *DeviceManagementService* in der Dienstliste des DNS-SD auftaucht, verbindet sich der *SystemManagementService* mit diesem Dienst und ruft dort die Device-Information (für die Geräteklasse), Device-Configuration (für die Geräte-ID) und Service-Informationen (für die Liste der verfügbaren Dienste) ab. Auf diese Weise kann der SystemManagementService

— eine Zuordnung von fachlicher Identifikation (Geräteklasse und Geräte-ID) zu technischer Identifikation (Dienstadressierung nach DNS-SD) machen und

— prüfen, ob alle erforderlichen Dienste auf den in der System-Konfiguration vorgesehenen Geräten vorhanden sind.

❖ **Establish Connection with the published Services of the Vehicle Operation Functionalities and Execution of functional Tasks**

The *SystemManagementService* is informed via DNS-SD, if a new service is added to the DNS-SD list of services. As soon as a *SystemDocumentationService* appears in this list, the *SystemManagementService* connects with this service and request the system configuration (in Abbildung 5, SDS/GetSystemConfiguration).

As soon as a *DeviceManagementService* appears in the list of services of the DNS-SD, the *SystemManagementService* connects with this service and retrieves device information (for the device class), device configuration (for the device ID), and service information (for the list of available services). This way, the SystemManagementService can

- Assign the functional identification (device class and device ID) to the technical identification (service addressing according to DNS-SD) and verify, whether
- All required services are available on the devices specified in the system configuration.

5.3.2 Stufe 2: Start der fachlichen Dienste / Stage 2: Start functional Services

5.3.2.1 Start weiterer Dienste

Der *Dienst X* auf Gerät A wird durch den *SystemManagementService* gestartet, indem die Dienst-Start-Operation des *DeviceManagementService* des Geräts A aufgerufen und der Name des Dienstes übergeben wird (vgl. Kapitel 7.1.23). Analog erfolgt der Start des *Dienstes Y* auf Gerät B über den *DeviceManagementService* von Gerät B.

❖ **Start of further Services**

Service X on device A is started by the *SystemManagementService* by calling the service start operation of the *DeviceManagementService* of device A and providing the name of the service (cf. chapter 7.1.23). *Service Y* on device B is analogously started via the *DeviceManagementService* on device B.

5.3.2.2 Veröffentlichung weiterer Dienste

Diese Stufe wird zeitlich versetzt durchlaufen. Jeder Dienst veröffentlicht sich, sobald er betriebsbereit ist. Die Dienste *Dienst X* und *Dienst Y* veröffentlichen sich jeweils, sobald sie betriebsbereit sind.

❖ Publish further Services

This stage is executed with a time offset. Every service is published, as soon as it is operational. Services *Service X* and *Service Y* are published, as soon as they are operational.

5.3.2.3 Verbindungsaufbau mit den veröffentlichten Diensten und Ausführung fachlicher Aufgaben

Der *SystemManagementService* fragt zyklisch den Status aller Dienste und Geräte bei den *DeviceManagementServices* der Geräte ab.

Im Beispiel richtet außerdem der *Dienst Y* beim *SystemManagementService* ein Abonnement ein, um kontinuierlich über Änderungen des Systemzustands informiert zu sein.

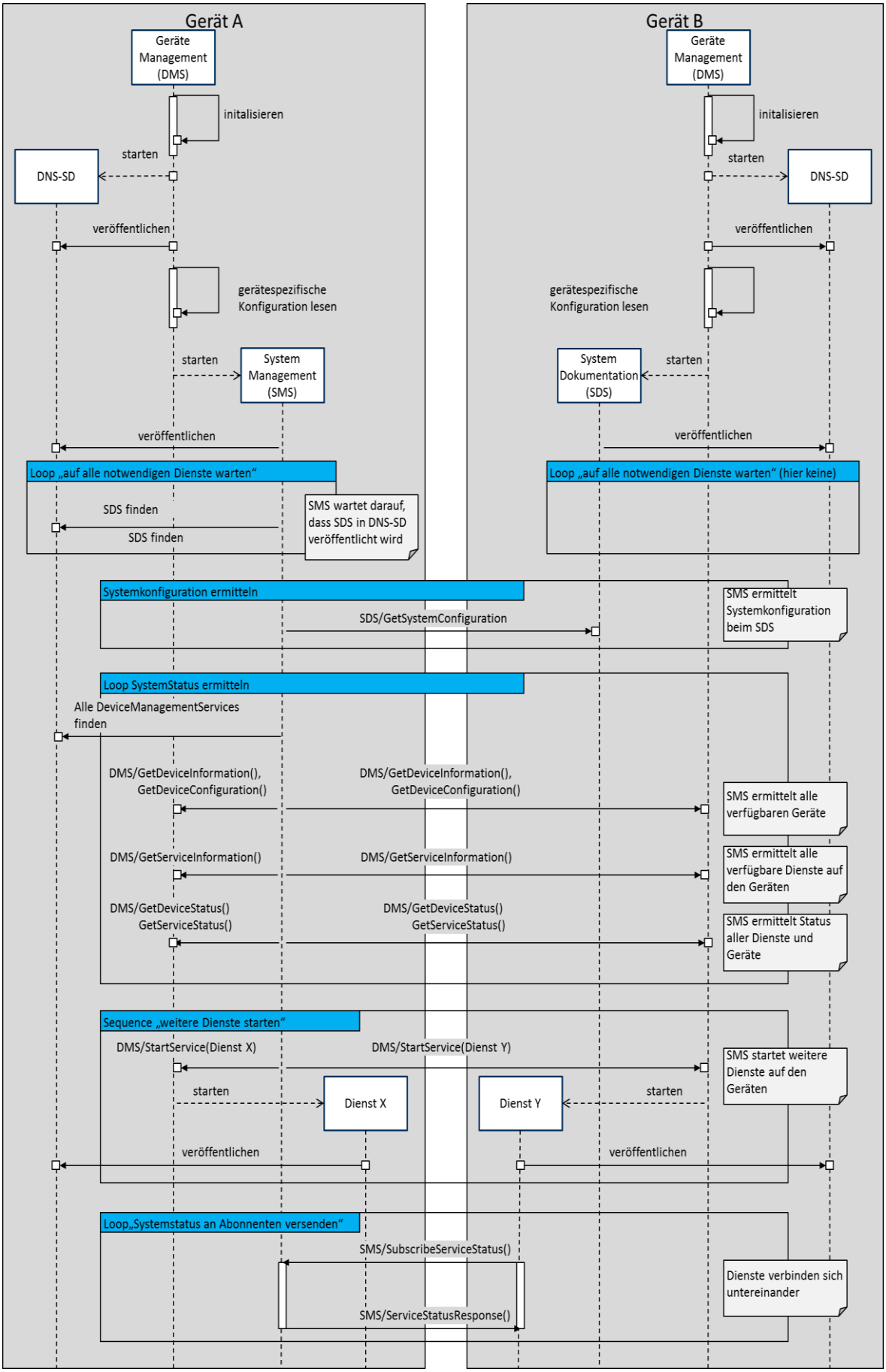

Abbildung 5 Beispielhafter Startup eines IBIS-IP-Systems

❖ Establish Connection with the published Services and Execution of functional Tasks

The `SystemManagementService` periodically requests the status of all services and devices from the `DeviceManagementServices` of the devices.

In the example, `Service Y` also establishes a subscription to `SystemManagementService` in order to be continuously informed about changes in the system state.

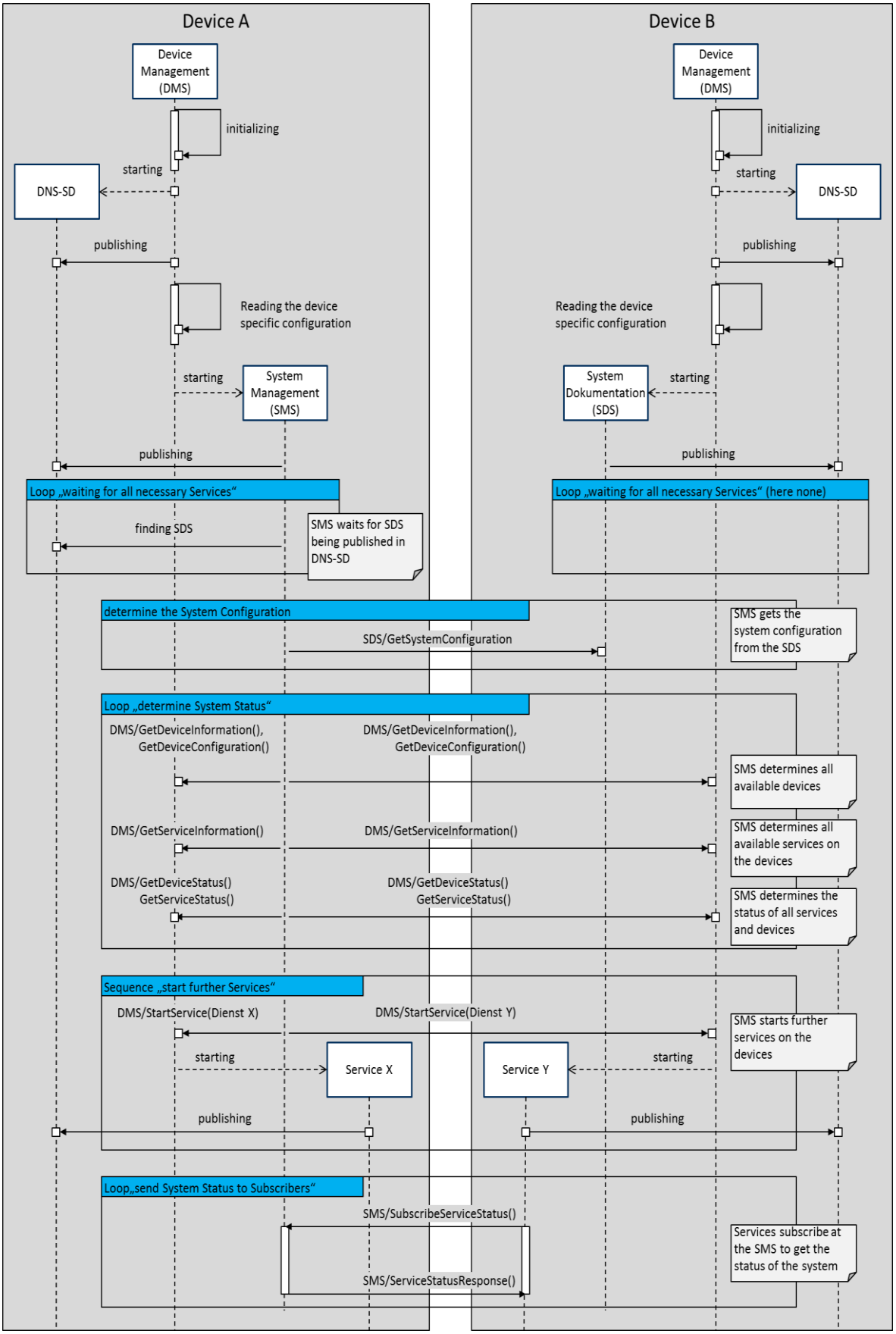

Figure 6: Example of the startup of an IBIS-IP system

6 Strukturierung der Informationsinhalte

In IBIS-IP werden Dateninhalte in Klartext übertragen. Das bedeutet insbesondere, dass mehrere Informationen nicht in einem Datenfeld kodiert übertragen werden dürfen, sondern entsprechend in mehrere Datenfelder einer Datenstruktur aufzubrechen sind.

Es wird weiterhin Wert auf die Verwendung etablierter und weit verbreiteter Verfahren zur Übertragung strukturierter Daten gelegt.

Informationsinhalte werden deshalb in IBIS-IP mit Hilfe von XML-Datenstrukturen übertragen und können mit Hilfe eines XML-Schemas (XSD = XML Schema Definition) entsprechend validiert werden. Neben der hier dargestellten Form stellt der VDV über seine Webseite die XSD-Dateien der spezifizierten Dienste kostenlos zur Verfügung.

❖ Structuring of the information contents

In IBIS-IP, data contents are transferred in plain text. This means in particular that multiple information must not be coded and transferred on one data field. It must be respectively broken down into several data fields of a data structure.

The use of established and commonly used methods for the structured data transfer is important.

Thus, information contents are transferred in IBIS-IP with the help of XML data structures, and can be respectively validated using a XML schema (XSD = XML schema definition). In addition to the form represented here, VDV provides the XSD files of the specified services free of charge on their webpage.

6.1 Notation der XML-Elemente und -Strukturen

Die in diesem Dokument vorgestellten IBIS-IP-Schnittstellen werden mit Hilfe von XML-Schema definiert. Die Objekte, die über die Schnittstelle ausgetauscht werden, liegen folglich als XML-Elemente vor. Die Beschreibung der XML-Elemente wird in diesem Dokument in einer Tabellenform vorgenommen, die aus SIRI (CEN, TS 15531 Part 1) stammt. Sie ist sehr kompakt, übersichtlich und bietet eine Vielzahl an strukturellen Informationen, die ansonsten nur in der XML-Schema-Definition sichtbar wird. Dieses Kapitel erläutert die Notation der Tabellenform, die ab Kapitel 7 genutzt wird.

Alle Namen von Elementen, Datentypen und Attributen sind in Englisch gehalten, um eine etwaige Normierung auf europäischer Ebene vorzubereiten und den Austausch mit europäischen Partnern zu erleichtern.

❖ Notation of XML Elements and Structures

The IBIS-IP interfaces presented in this document are defined with the help of a XML schema. Thus, the objects that are exchanged via the interface are available as XML elements. In this document, the XML elements are described in tabular form that originates from SIRI (CEN, TS 15531 Part 1). It is very compact, clear and offer numerous structural information, which is otherwise only visible on the XML schema definition. This chapter explains the notation of the table form used starting from chapter 7.

All names of elements, data types, and attributes are in English to prepare a possible standardization at the European level and to facilitate the exchange with European partners.

6.1.1 Darstellung von XML-Elementen im Text

Eine konsistente Notation der XML-Elemente soll helfen, technisch wichtige Information beim Lesen bereit zu stellen.

— XML-Elemente werden in Groß-Klein-Schreibweise (Upper Camel Case) fett und kursiv geschrieben, z. B.: ***VehicleJourneyRef***. Die Elementnamen sind – wo immer möglich und sinnvoll – an Begriffe aus TransModel angelehnt. Fehlt in TransModel ein geeigneter Begriff für ein Konzept oder Objekt, so wurde versucht, den entsprechenden Begriff aus JourneyWeb oder das passende Konzept aus DELFI zu übernehmen.

— Datentypen werden kursiv dargestellt, z. B: *xsd:boolean.*

— Code-Beispiele werden in kleinerer Schrift wiedergegeben.

❖ Representation of XML Elements in Text

A consistent notation of the XML elements should facilitate the provision of technically important information when reading.

- XML elements are written in upper-lower-case notation (Upper Camel Case), bold, and italic, e.g. ***VehicleJourneyRef***. Whenever possible and meaningful, the element names are based on terms from TransModel. If there is no suitable term available in TransModel for a concept or object, it was attempted to apply the respective term from JourneyWeb or the suitable concept from DELFI.
- Data types are written in italic, e.g. *xsd:boolean.*
- Code examples are presented in smaller font.

6.1.2 Tabellennotation für Operationen

Darüber hinaus werden Operationen eines Dienstes in einer Tabelle der folgenden Form beschrieben.

Operation	**Request/ Response**	**Verwendeter Datentyp, Datenstruktur**
GetData	Req.	-
	Resp.	BeaconLocationService. GetDataResponseStructure
SubscribeData	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeData	Req.	TerminateSubscribeRequestStructure
	Resp.	TerminateSubscribeResponseStructure

Tabelle 7 Beispiel für Operationsnotation in IBIS-IP

❖ Table Notations for Operations

Beyond that, operations of a service are describe in a table of the following form.

Operation	**Request/ Response**	**Data type, data structure used**
GetData	Req.	-
	Resp.	BeaconLocationService. GetDataResponseStructure
SubscribeData	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeData	Req.	TerminateSubscribeRequestStructure
	Resp.	TerminateSubscribeResponseStructure

Table 8: Example of an operation notation in IBIS-IP

6.1.3 Tabellennotation von XML-Strukturen

In dieser Schrift werden XML-Strukturen in einer Tabellennotation dargestellt (vgl. Tabelle 9). Für jedes wichtige Dienst-Anfrage/Antwort-Element findet sich eine eigene Tabelle. Weitere Tabellen werden für alle wesentlichen Kind-Elemente, aus denen die komplexen Strukturen aufgebaut sind, angegeben. Um Platz zu sparen, werden die Spaltenüberschriften nur im Beispiel in Tabelle 9 angezeigt und bei allen folgenden Tabellen nicht wiederholt. In den Tabellen wird ein konsistenter Satz an Regeln zur Beschreibung der XML-Elemente und der daran geknüpften Bedingungen verwendet.

Gruppierung	**Elementname**	**Min : Max**	**Datentyp**	**Erläuterung**
ContinuousServiceStructure			*+Structure*	Eine Fahrgastbewegung mit Hilfe eines kontinuierlichen, nicht fahrplangebundenen Verkehrsmittels.
	a ***ContinuousMode***	**-1:1**	*walk | parkAndRide | demandResponsive*	Modalität für kontinuierliche Verkehre
	b ***IndividualMode***		*walk | cycle | taxi | self-drive-car | others-drive-car | motorcycle | truck*	Verkehrsmittelmodalität für Individualverkehr
DatedService	***OperatingDay***	**1:1**	*→OperatingDay*	Betriebstag der Fahrt.
	VehicleRef	0:1	*→Vehicle*	Fahrzeug-ID.
ServiceJourney	***JourneyRef***	**1:1**	*→Journey*	Fahrt-ID. .
LineIdentity	***LineRef***	**1:1**	*→Line*	Linien-ID. .
	DirectionRef	**1:1**	*→Direction*	Richtungs-ID..
Service	***Mode***	**1:1**	*+Mode*	Verkehrsmitteltyp. .
	PublishedLineName	**1:1**	*InternationalText*	Liniennummer oder -name, wie in der Öffentlichkeit bekannt.
	OperatorRef	0:1	*→Operator*	Operator-ID. .
	RouteDescription	0:1	*InternationalText*	Beschreibung des Fahrwegs.
	Via	0:*	*+ServiceViaPoint*	Wichtige Halte auf dem Fahrweg.
	Attribute	0:*	*+GeneralAttribute*	Hinweise und Attribute (mit Klassifikationen) zur Fahrt.
ServiceOrigin	*OriginStopPointRef*	0:1	*→StopPoint*	ID des ersten Haltepunkts der Fahrt; Starthaltestelle.
	OriginText	0:1	*InternationalText*	Name des ersten Haltepunkts der Fahrt, der Starthaltestelle.

Tabelle 9 Beispiel für Kapitel 7 für die tabellarische Notation einer XML-Struktur

❖ Table Notation of XML Structures

In this document, XML structures are represented in a table notation (cf. Tabelle 9). There is a separate table for every important service request/response element. Further tables are specified for all important child elements, which are used in structures that are more complex. In order to save space, the column headers are only shown in the example in Tabelle 9. They will not be repeated in any of the following tables. A consistent set of rules is used in the tables in order to describe the XML elements and the associated conditions.

Grouping	Element name	Min : Max	Data type	Explanation
ContinuousServiceStructure			*+Structure*	A passenger movement with the help of a continuous means of transportation not bound to a schedule.
	a ***ContinuousMode***	**-1:1**	*walk | parkAndRide | demandResponsive*	Modality for continuous traffic
	b ***IndividualMode***		*walk | cycle | taxi | self-drive-car | others-drive-car | motorcycle | truck*	Modality of means of traffic for individual traffic
DatedService	***OperatingDay***	**1:1**	*→OperatingDay*	Operating day of the journey
	VehicleRef	0:1	*→Vehicle*	Vehicle ID.
ServiceJourney	***JourneyRef***	**1:1**	*→Journey*	Journey ID. .
LineIdentity	***LineRef***	**1:1**	*→Line*	Line ID. .
	DirectionRef	**1:1**	*→Direction*	Direction ID.
Service	***Mode***	**1:1**	*+Mode*	Type of means of transportation. .
	PublishedLineName	**1:1**	*InternationalText*	Line number or name as known by the public.
	OperatorRef	0:1	*→Operator*	Operator ID. .
	RouteDescription	0:1	*InternationalText*	Description of the route.
	Via	0:*	*+ServiceViaPoint*	Important stops on the route.
	Attribute	0:*	*+GeneralAttribute*	Notes and attributes (with classifications) regarding the journey.
ServiceOrigin	*OriginStopPointRef*	0:1	*→StopPoint*	ID of the first stop point of the journey, i.e. point of departure.
	OriginText	0:1	*InternationalText*	Name of the first stop point of the journey, i.e. point of departure.

Table 10: Example for chapter 7 for the tabularized notation of a XML structure

6.1.3.1 Gruppierung

In der ersten Spalte befindet sich gelegentlich ein Bezeichner, der die Elemente in sinnvolle Gruppierungen einteilt, z. B. *Service* oder *ServiceOrigin*. Dies dient rein zu Dokumentationszwecken und entspricht in den meisten Fällen den Namen einer XML-Gruppe, die im XML-Schema verwendet wurde. Die Verwendung von Gruppierungen hat nur den Zweck, die Elemente zu organisieren und damit für mehr Klarheit und bessere Wiederverwendbarkeit zu sorgen.

❖ Grouping

There is sometimes an identifier in the first column, which classifies the elements into meaningful groupings, e.g. *Service* or *ServiceOrigin*. This is purely used for documentation purposes only. In most cases, this corresponds to the names of a XML group used in the XML schema. Groupings are used to organize the elements and thus, to provide more clarity and better reusability.

6.1.3.2 Elementname

Elementnamen werden kursiv in der zweiten Spalte wiedergegeben, z. B. *OperatingDay*. Handelt es sich um ein verpflichtendes Element, so wird es **fett** gedruckt. Optionale Elemente werden nicht fett gedruckt. Der Name der Struktur selbst ist links oben in der Tabelle angegeben.

Elemente, die geerbt (XML: “derived by extension”) oder anonym verwendet werden, tragen im Namensfeld drei Doppelpunkte “:::” zur Kennzeichnung.

❖ Element Name

Element names are shown in italic in the second column, e.g. *OperatingDay*. If the element is a mandatory element, it is printed in **bold**. Optional elements are not printed in bold. The name of the structure can be found in the top left of the table.

Elements, which are derived (XML" “derived by extension”) or used anonymously, are marked in the name field with three colons “:::”.

6.1.3.3 Multiplizität & Choice (Min:Max)

Die Bedingungen, ob ein Element verpflichtend oder optional ist oder ob es einfach oder mehrfach innerhalb des übergeordneten Elements auftreten kann, werden in der dritten Spalte Min:Max angegeben. Dabei werden die üblichen UML-Konventionen „min:max“ angewendet, so steht z. B. „0:1“ für ein optionales, einfaches Element, „**1:1**“ zeigt ein verpflichtendes, einfaches Element an, „0:*“ steht für ein optionales, mehrfaches Element usw. Verpflichtende Elemente werden **fett** gedruckt.

In manchen Fällen muss ein Element aus seiner Menge ausgewählt werden (XML-Choice). Dies wird durch ein vorangestelltes Minuszeichen symbolisiert, z. B. „**-1:1**“. In diesem Fall steht vor dem Elementnamen noch ein Kleinbuchstabe, der die Auflistung der Wahlmöglichkeiten anzeigt. Bei optionalen Auswahlmöglichkeiten (Choices) steht im Min-Wert eine Null: „-0:1“.

❖ Multiplicity & Choice (Min:Max)

The conditions, whether an element is mandatory or optional, or whether it can occur one or several times within the superordinate element, are indicated in the third column Min:Max. In this context, the common "min:max:" UML conventions are used. E.g. "0:1" stands for an optional, simple element, "**1:1**" indicates a mandatory, simple element. "0:*" stands for an optional, multiple element etc. Mandatory elements are printed in **bold**.

In some cases, an element must be selected from its set (XML choice). This is indicated by a prefixed minus sign, e.g. "**-1:1**". In this case, the element name is prefixed by a lower case letter indicating the list of selection possibilities. In the case of optional selection options (choices), a zero is contained in the min value: "-0:1".

6.1.3.4 Datentyp

Die Datentypen werden in der vierten Spalte kursiv angegeben, z. B. *InternationalText*. Falls der Namensraum (namespace) vom Standard XML Namensraum abweicht, wird er mit angegeben, z. B. „*xs:dateTime*“ oder „*siri:PtSituationElement*“.

— Ein komplexer Datentyp, der selbst Strukturen als Kind-Elemente enthält, wird in der Spalte Datentyp mit „*+Structure*“ gekennzeichnet.

— Wo Elemente als Referenzen (Fremdschlüssel) auf andere Objekte verwendet werden, wird als Datentyp der Typ des referenzierten Objekts mit vorangestelltem Pfeil verwendet. Zum Beispiel „ *→StopPoint*“ als Typ einer Referenz (*StopPointRefStructure*) auf ein Objekt vom Typ „StopPointType“.

— Aufzählungstypen (Enumerated types) werden an den meisten Stellen unmittelbar mit den verwendbaren Werten dargestellt, z. B. „*walk | cycle*“. Nur in einigen Fällen mit sehr umfangreichen Aufzählungen, die an mehreren Stellen wiederverwendet werden, wird ein Typ deklariert und referenziert.

— Um Platz zu sparen, werden bei der Angabe der Datentypen Abkürzungen verwendet. Z. B. wird auf die Endungen “Structure” und “Type” durchgehend verzichtet. Statt beispielsweise „*InternationalTextStructure*“ wird also immer „*InternationalText*“ als Datentyp angegeben.

❖ Data Type

The data types are indicated in italic in the fourth column, e.g. *InternationalText*. If the namespace deviates from the standard XML namespace, it is indicated as well, e.g. "*xs:dateTime*" or "*siri:PtSituationElement*".

- A complex data type, which contains structures as child elements, is marked with "*+Structure*" in the Data type column.
- If elements are used as references (foreign keys) to other objects, the type of the referenced element with a prefixed arrow is used as data type. For example, " *→StopPoint*" as type of a reference (*StopPointRefStructure*) to an object of type "StopPointType".
- In the most cases, enumerated types are immediately represented with the usable values, e.g. "*walk | cycle*". A type is only declared and referenced in some cases with very extensive enumerations, which are used at several locations.
- In order to save space, abbreviations are used for the data types. E.g., the endings "Structure" and "Type" are always omitted. E.g., "*InternationalText*" is always used instead of "*InternationalTextStructure*".

6.1.3.5 Erläuterung

Alle Elemente erhalten in der letzten Spalte eine Erläuterung ihres Verwendungszwecks. An vielen Stellen wird auf weitere Stellen im Text hingewiesen, so z. B. bei komplexen Kind-Elementen an die Stelle, wo ihre Tabellenbeschreibung zu finden ist. An einigen Stellen ist die Erläuterung zu umfangreich und würde die Tabellenform sprengen. Dann finden sich diese Anmerkungen im Text unterhalb der Tabelle.

Ergänzend zu diesem Dokument existiert eine XSD-Datei mit den Inhalten in digitaler Form. Diese steht, falls nicht mit diesem Dokument mitgeliefert, zum Download unter www.vdv.de/ip-kom-oev.aspx zur Verfügung.

❖ Explanation

The last column contains for all elements an explanation of their purpose. In many cases, it is referenced to other locations in the text. E.g. in the case of complex child elements, the location is referenced, where their table definition can be found. In some cases, the explanation is too extensive and would explode the tabular form. In these cases, the remarks can be found in the text below the table.

A XSD file with the contents in digital form exists as supplement to this document. If not enclosed with this document, this file can be downloaded under www.vdv.de/ip-kom-oev.aspx.

7 Beschreibung einzelner Fachdienste

7.1 Dienst DeviceManagementService

Der Dienst `DeviceManagementService` ist der Dienst, welcher die Aufgaben der Fachkomponente Geräte-Management innerhalb von IBIS-IP umsetzt.

Der Dienst ist für alle Daten als http-Dienst spezifiziert.

Dieser Dienst ist auf jedem an IBIS-IP teilnehmenden Gerät einmal vorhanden und wird mit dem Ausführen der Applikation auf dem Gerät automatisch gestartet.

❖ DeviceManagementService

The `DeviceManagementService` is a service which is used to maintain and manage the plugged devices.

As described in chapter 5.2.1 a DeviceManagementService has to run on each device participating in the IBIS-IP network.

The `DeviceManagementService` is designed as an http-service.

7.1.1 Operationen des DeviceManagementService

Die vom `DeviceManagementService` angebotenen Operationen lassen sich unterteilen in Operationen die das Gerät betreffen

— Statische Informationen über das Gerät abfragen

— Konfiguration des Geräts ändern

— Konfiguration des Geräts abfragen

— Gerät restarten, deaktivieren, aktivieren

— Status des Geräts abfragen

— Fehlermeldungen abfragen

sowie Operationen, die die Dienste auf dem Gerät betreffen

— Statische Informationen über die Dienste

— Dienste starten/stoppen/restarten

— Zustand der Dienste abfragen

❖ Operations of DeviceManagementService

The `DeviceManagementService` Operations are defined in tasks dealing on the one side with the device functions like

— Request static information about the device

— Change the device configuration

— Request the device configuration

— Restart, deactivate and activate the device

— Request the current device status

— Request error codes from the device

and on the other side with functions for Services on the device like

— Static information about the services

— Start, stop and restart services

— Request the state of the services

Operation	**Request/ Response**	**Verwendeter Datentyp, Datenstruktur / Used Datatype, Data structure**
GetDeviceInformation	Req.	-
	Resp.	DeviceManagementService. GetDeviceInformationResponseStructure
SubscribeDeviceInformation	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeDeviceInformation	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetDeviceConfiguration	Req.	-
	Resp.	DeviceManagementService. GetDeviceConfigurationResponseStructure
SetDeviceConfiguration	Req.	DeviceManagementService. SetDeviceConfigurationRequestStructure
	Resp.	DataAcceptedResponseStructure
SubscribeDeviceConfiguration	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeDeviceConfiguration	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetDeviceStatus	Req.	-
	Resp.	DeviceManagementService. GetDeviceStatusResponseStructure
SubscribeDeviceStatus	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeDeviceStatus	Req.	UnsubscribeRequestStructure

Operation	Request/ Response	Verwendeter Datentyp, Datenstruktur / Used Datatype, Data structure
	Resp.	UnsubscribeResponseStructure
GetDeviceErrorMessages	Req.	-
	Resp.	DeviceManagementService. GetDeviceErrorMessagesResponseStructure
SubscribeDeviceErrorMessages	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeDeviceErrorMessages	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
RestartDevice	Req.	-
	Resp.	DataAcceptedResponseStructure
DeactivateDevice	Req.	-
	Resp.	DataAcceptedResponseStructure
ActivateDevice	Req.	-
	Resp.	DataAcceptedResponseStructure
GetServiceInformation	Req.	-
	Resp.	DeviceManagementService. GetServiceInformationResponseStructure
SubscribeServiceInformation	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeServiceInformation	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetServiceStatus	Req.	-
	Resp.	DeviceManagementService. GetServiceStatusResponseStructure
SubscribeServiceStatus	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeServiceStatus	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
StartService	Req.	DeviceManagementService. StartServiceRequestStructure

Operation	Request/ Response	Verwendeter Datentyp, Datenstruktur / Used Datatype, Data structure
	Resp.	DataAcceptedResponseStructure
StopService	Req.	DeviceManagementService. StopServiceRequestStructure
	Resp.	DataAcceptedResponseStructure
RestartService	Req.	DeviceManagementService. RestartServiceRequestStructure
	Resp.	DataAcceptedResponseStructure
GetAllSubdeviceInformation	Req.	-
	Resp.	DeviceManagementService. GetAllSubdeviceInformationResponseStructure
SubscribeAllSubdeviceInformation	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeAllSubdeviceInformation	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetDeviceStatusInformation	Req.	-
	Resp.	DeviceManagementService. GetDeviceStatusInformationResponseStructure
SubscribeDeviceStatusInformation	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeDeviceStatusInformation	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetAllSubdeviceStatusInformation	Req.	-
	Resp.	DeviceManagementService. GetAllSubdeviceStatusInformationResponseStructure
Subscribe AllSubdeviceStatusInformation	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
Unsubscribe AllSubdeviceStatusInformation	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetAllSubdeviceErrorMessages	Req.	-
	Resp.	DeviceManagementService. GetAllSubdeviceErrorMessagesResponseStructure
SubscribeAllSubdeviceErrorMessages	Req.	SubscribeRequestStructure

Operation	Request/ Response	Verwendeter Datentyp, Datenstruktur / Used Datatype, Data structure
	Resp.	SubscribeResponseStructure
Unsubscribe AllSubdeviceErrorMessages	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
InstallUpdate	Req.	DeviceManagementService. InstallUpdateRequestStructure
	Resp.	DeviceManagementService. InstallUpdateResponseStructure
RetrieveUpdateState	Req.	DeviceManagementService. RetrieveUpdateStateRequestStructure
	Resp.	DeviceManagementService. RetrieveUpdateStateResponseStructure
GetUpdateHistory	Req.	-
	Resp.	DeviceManagementService. GetUpdateHistoryResponseStructure
FinalizeUpdate	Req.	DeviceManagementService. FinalizeUpdateRequestStructure
	Resp.	DeviceManagementService. FinalizeUpdateResponseStructure
FinalizeAllPendingUpdates	Req.	-
	Resp.	DataAcceptedResponseStructure

Tabelle 11 Beschreibung von Operationen des DeviceManagementService / Description of Operationen des DeviceManagementService

7.1.2 Aktualisierung von Firmware und Konfiguration

Eine Aktualisierung wird mit der Operationen InstallUpdate eingeleitet. Mit Hilfe dieser Operation kann ein Update-Controller einem IBIS-IP-Gerät eine Update-Datei übergeben, die dieses dann installiert. Der Speicherort der Update-Datei wird als URI übergeben. Das Gerät lädt die Datei herunter und prüft anschließend, ob diese sich für eine Aktualisierung eignet. Wenn das nicht der Fall ist, wird die Aktualisierung zurückgewiesen. Die Operation RetrieveUpdateState ist einzusetzen, um bei einer laufenden Aktualisierung deren aktuellen Status abzufragen. Nach dem Abschluss einer Aktualisierung ist diese mit der Operation FinalizeUpdate zu finalisieren.

Das IBIS-IP-Gerät speichert persistent eine Update-History. Diese kann mit Hilfe der Operation GetUpdateHistory abgefragt werden.

Abhängigkeiten von anderen Aktualisierungen können nicht verwaltet werden. Wenn unterschiedliche Komponenten eines IBIS-IP-Geräts zur selben Zeit passende Aktualisierungen erhalten müssen, sind diese in einer einzigen Update-Datei zusammenzufassen.

Der Update-Controller unterstützt keine Aktualisierungspläne.

Die richtige Zuordnung einer Update-Datei zu einem Gerät ist nicht die Aufgabe des Update-Controllers. Dafür ist der Update-Manager zuständig, der den Update-Controller bei Bedarf

aufruft. Der Update-Controller gibt das Ergebnis einer Aktualisierung an den Update-Manager weiter. Dieser wertet es entsprechend aus.

Der Update-Controller muss eine Software sein und ist für diese Aufgaben zuständig:

- Steuerung des Update-Prozesses beginnend mit der Operation InstallUpdate und endend mit der Operation FinalizeUpdate
- Übergabe des Ergebnisses eines Updates an den Update-Manager

Der Update-Manager kann eine Software sein oder eine Person. Er ist für diese Aufgaben zuständig:

- Auswahl der Update-Datei
- Planung von Aktualisierungen

Der Update-Controller startet das Gerät neu, wenn es den Aktualisierungsstatus *DeviceRestartRequired* zurückmeldet. Mindestens nach dem Abschluss einer Aktualisierung ist das betreffende Gerät neu zu starten. Neustarts im Verlauf des Update-Prozesses müssen wie erforderlich ausgeführt werden. Der initiale Aktualisierungsstatus nach einem solchen Neustart muss *UpdateRunning* sein.

❖ Update of Firmware and Configuration

Update is initiated by operation InstallUpdate. Using this operation an update controller provides an update file for an IBIS-IP device which can be used by the device for installation. Location of the update file is given by an URI. Device downloads update file and checks it afterwards. Update is rejected if update file is not suitable. Operation RetrieveUpdateState has to be used during running update to get its current state. After completion of update it has to be finalized using operation FinalizeUpdate.

Update history is persistently stored by IBIS-IP device. It can be queried using operation GetUpdateHistory.

Management of dependencies to other updates is not covered. If several components of an IBIS-IP device must receive different updates at the same time, these updates must be included in a single update file.

Update controller does not support update schedules.

Right assignment of update file to the device to be updated is not the task of the update controller but of an update manager which calls the update controller as needed. Update controller reports the update result to the update manager which handles the result appropriately.

The update controller has to be a program. These tasks are assigned to it:

- Controlling of update process beginning with operation InstallUpdate and ending with operation FinalizeUpdate
- Reporting update result to update manager

The update manager may be a program or a person. These tasks are assigned to it:

- Selection of update file

- Scheduling of updates

The update controller restarts device if it returns update status *DeviceRestartRequired*. At least at the end of an update a restart of the updated device is required. Intermediate restarts shall be done as needed. Initial update status after an intermediate restart shall be *UpdateRunning*.

7.1.2.1 Update-Prozess

1. Der Update-Controller führt die Operation *InstallUpdate* aus und setzt seine Tätigkeit fort, wenn das Gerät die Aktualisierung durch Rückgabe des Werts *UpdateAccepted* akzeptiert.
2. Mit Hilfe der Operation *RetrieveUpdateState* wird periodisch der Aktualisierungsstatus abgefragt. Das maximale Abfrageintervall beträgt 5s. Der minimale Wert ist 1s.
3. Das Gerät wird mittels der Operation *DeviceManagementService*.*RestartDevice* neu gestartet, wenn es den Aktualisierungsstatus *DeviceRestartRequired* zurückgibt.
4. Der Update-Controller wartet während das Gerät herunterfährt. Der Standardwert der Wartezeit ist 10s.
5. Der Update-Controller prüft, ob das Gerät wieder mit dem IBIS-IP-Netzwerk verbunden ist, und kehrt danach zur Abfrage des Aktualisierungsstatus zurück. Der Standardwert der Wartezeit auf das Wiederverbinden ist 5 Minuten.
6. Die Aktualisierung wird durch die Operation *FinalizeUpdate* finalisiert, wenn das Gerät einen anderen Aktualisierungsstatus als *UpdateRunning* oder *DeviceRestartRequired* zurückgibt. Die Aktualisierung ist zu wiederholen, wenn der finale Aktualisierungsstatus *DownloadUpdateFileFailed* ist.
7. Der Update-Controller übermittelt das Ergebnis der Aktualisierung an den Update-Manager.

❖ Update Process

1. Update controller performs operation *InstallUpdate* and continues if the device accepts the update by returning *UpdateAccepted*.
2. Query update state periodically using operation *RetrieveUpdateState*.
   Maximum query interval is 5s. Minimum query interval is 1s.
3. Restart device using operation *DeviceManagementService*.*RestartDevice* if the device returns update status *DeviceRestartRequired*.
4. Wait while the device shuts down. The default value of the shut down wait time is 10s.
5. Check if the device is reconnected to the IBIS-IP network and go to update state query afterwards. The default value of the reconnect wait time is 5 minutes.
6. Finalize the update using operation *FinalizeUpdate* if the device returns any other update status than *UpdateRunning* or *DeviceRestartRequired*. Repeat the update if the final update status is *DownloadUpdateFileFailed*.
7. Report the update result to the update manager.

7.1.3 Data Structure of Operation GetDeviceInformation

The operation GetDeviceInformation returns static device information like serial number, manufacturer etc.

7.1.3.1 Request

Because of being a ***GetDeviceInformation*** operation, there is no request structure for this operation.

7.1.3.2 Response

DeviceManagementService.GetDeviceInformationResponse				*+Structure*	**Request structure with non configuration able device information**
				choice	One of the choices below
	a	***DeviceManagementService.GetDeviceInformationResponseData***	**–1:1**	*+DeviceManagementService.GetDeviceInformationResponseData*	Detailed request structure with the non configurable device parameters (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 12 Description of DeviceManagementService.GetDeviceInformationResponse

DeviceManagementService.GetDeviceInformationResponseData			*+Structure*	**Detailed response structure with static device settings**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	DeviceInformation	**1:1**	*+DeviceInformation*	Detailed response structure (cf. VDV 301-2-1)

Table 13 Description of DeviceManagementService.GetDeviceInformationResponseData

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.4 Data Structure of Operation SubscribeDeviceInformation

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.5 Data Structure of Operation UnsubscribeDeviceInformation

For the subscription the data structures from VDV 301-2-1 are used.

7.1.6 Data Structure of Operation GetDeviceConfiguration

The **GetDeviceConfiguration** operation enables to set the single variable parameter of a device. This parameter is the device –ID, which is used as a reference to the plugged position inside the vehicle

7.1.6.1 Request

Because of being a ***GetDeviceConfiguration*** operation, there is no request structure for this operation.

7.1.6.2 Response

DeviceManagementService.GetDeviceConfigurationResponse				*+Structure*	**Response structure with device configuration data content**
				choice	One of the structures below
	a	***DeviceManagementService.GetDeviceConfigurationResponseData***	**–1:1**	*+DeviceManagementService.GetDeviceConfigurationResponseData*	Detailed response structure with the device configuration (cf. below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 14 Description of DeviceManagementService.GetDeviceConfigurationResponse

DeviceManagementService.GetDeviceConfigurationResponseData			*+Structure*	**Detailed response structure with the device configuration**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	DeviceID	**1:1**	*IBIS-IP.int*	Device-ID (device plug-in position)

Table 15 Description of DeviceManagementService.GetDeviceConfigurationResponseData

7.1.7 Data Structure of Operation SetDeviceConfiguration

With this operation it is possible to set the device-ID. This setting happens just the first time after the device is plugged to the system and is stored in the device configuration file (proprietary). This fall back solution is needed when the device is not able to set this ID based on certain automatic parameters.

7.1.7.1 Request

DeviceManagementService.SetDeviceConfigurationRequest			*+Structure*	**Request structure which enables the setting of the device ID in the device configuration**
	DeviceID	**1:1**	*IBIS-IP.int*	Value of the device-ID

Table 16 Description of DeviceManagementService.SetDeviceConfigurationRequest

7.1.7.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf. VDV 301-2-1) is used.

7.1.8 Data Structure of Operation SubscribeDeviceConfiguration

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.9 Data Structure of Operation UnsubscribeDeviceConfiguration

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.10 Data Structure of Operation GetDeviceStatus

7.1.10.1 Request

Because of being a ***GetDeviceStatus*** operation, there is no request structure for this operation.

7.1.10.2 Response

DeviceManagementService.GetDeviceStatusResponse				*+Structure*	**Response structure with the device status**
				choice	One of the structures below
	a	***DeviceManagementService.GetDeviceStatusResponseData***	**–1:1**	*+DeviceManagementService.GetDeviceStatusResponseData*	Detailed response structure with the device status information (cf. below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 17 Description of DeviceManagementService.GetDeviceStatusResponse

DeviceManagementService.GetDeviceStatusResponseData			*+Structure*	**Detailed response structure with the device status information**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	DeviceState	**1:1**	*DeviceStateEnumeration*	Device status (cf. VDV 301-2-1)

Table 18 Description of DeviceManagementService.GetDeviceStatusResponseData

7.1.11 Data Structure of Operation SubscribeDeviceStatus

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.12 Data Structure of Operation UnsubscribeDeviceStatus

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.13 Data Structure of Operation GetDeviceErrorMessages

7.1.13.1 Request

Because of being a ***GetDeviceErrorMessage*** operation, there is no request structure for this operation.

7.1.13.2 Response

DeviceManagementService.GetDeviceErrorMessagesResponse				*+Structure*	**Response structure for device error messages**
				choice	One of the choices below
	a	***DeviceManagementService.GetDeviceErrorMessagesResponseData***	**–1:1**	*+DeviceManagementService.GetDeviceErrorMessagesResponseData*	Detailed response structure for device errors
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 19 Description of DeviceManagementService.GetDeviceErrorMessagesResponse

DeviceManagementService.GetDeviceErrorMessagesResponseData			*+Structure*	**Detailed response structure for device errors**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	ErrorMessage	**10:***	*+ Message*	Error message

Table 20 Description of DeviceManagementService.GetDeviceErrorMessagesResponseData

7.1.14 Data Structure of Operation SubscribeDeviceErrorMessages

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.15 Data Structure of Operation UnsubscribeDeviceErrorMessages

The data structures described in VDV 301-2-1 are used to set up subscriptions.

7.1.16 Data Structure of Operation RestartDevice

7.1.16.1 Request

With the request **`RestartDevice`** is no data transmitted.

7.1.16.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf. VDV 301-2-1) is used.

7.1.17 Data Structure of Operation DeActivateDevice

7.1.17.1 Request

With the request ***`DeactivateDevice`*** is no data transmitted.

7.1.17.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf. VDV 301-2-1) is used.

7.1.18 Data Structure of Operation ActivateDevice

7.1.18.1 Request

With the request ***ActivateDevice*** is no data transmitted.

7.1.18.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf. VDV 301-2-1) is used.

7.1.19 Data Structure of Operation GetServiceInformation

7.1.19.1 Request

Because of being a ***GetServiceInformation*** operation, there is no request structure for this operation.

7.1.19.2 Response

DeviceManagementService.GetServiceInformationResponse				*+Structure*	**Response structure with information about the available services at the device**
				choice	One of the choices below
	a	***DeviceManagementService.GetServiceInformationResponseData***	**–1:1**	*+DeviceManagementService.GetServiceInformationResponseData*	Detailed response structure with information about the available services (cf. below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 21 Description of DeviceManagementService.GetServiceInformationResponse

DeviceManagementService.GetServiceInformationResponseData			*+Structure*	**Detailed response structure with information about the available services**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	ServiceInformationList	**1:1**	*+ServiceInformationList*	List of available services (cf. VDV 301-2-1)

Table 22 Description of DeviceManagementService.GetServiceInformationResponseData

7.1.20 Data Structure of Operation SubscribeServiceInformation

For this subscription the data structures from VDV 301-2-1 are used.

7.1.21 Data Structure of Operation UnsubscribeServiceInformation

To terminate this subscription the structures of VDV 301-2-1 are used.

7.1.22 Data Structure of Operation GetServiceStatus

7.1.22.1 Request

Because of being a ***GetServiceStatus*** operation, there is no request structure for this operation.

7.1.22.2 Response

DeviceManagementService.GetServiceStatusResponse				*+Structure*	**Response structure with status information about the services located at the device**
				choice	One of the choices below
	a	***DeviceManagementService.GetServiceStatusResponseData***	**–1:1**	*+DeviceManagementService.GetServiceStatusResponseData*	Detailed response structure with the status of services running on the device (cf. below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 23 Description of DeviceManagementService.GetServiceStatusResponse

DeviceManagementService.GetServiceStatusResponseData			*+Structure*	**Detailed response structure with the services located at the device**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	ServiceSpecificationWithStateList	**1:1**	*+ServiceSpecificationWithStateList*	Service list including the status (cf. VDV 301-2-1)

Table 24 Description of DeviceManagementService.GetServiceStatusResponseData

7.1.23 Data Structure of Operation SubscribeServiceStatus

For this subscription the data structures from chapters VDV 301-2-1 are used.

7.1.24 Data Structure of Operation UnsubscribeServiceStatus

To terminate this subscription the structures of chapters VDV 301-2-1 are used.

7.1.25 Data Structure of Operation StartService

7.1.25.1 Request

DeviceManagementService.StartServiceRequestStructure			*+Structure*	**Request structure which enables to start one specific (in the ServiceSpecification specified) service**
	ServiceSpecification	**1:1**	*+ServiceSpecification*	Reference on the services to be started (cf. VDV 301-2-1)

Table 25 Description of DeviceManagementService.StartServiceRequestStructure

7.1.25.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf. VDV 301-2-1) is used.

7.1.26 Data Structure of Operation StopService

7.1.26.1 Request

DeviceManagementService.StopServiceRequestStructure			*+Structure*	**Request structure which enables to stop one specific (in the ServiceSpecification specified) service.**
	ServiceSpecification	**1:1**	*+ServiceSpecification*	Reference at the stopped services (cf. VDV 301-2-1)

Table 26 Description of DeviceManagementService.StopServiceRequestStructure

7.1.26.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf. VDV 301-2-1) is used.

7.1.27 Data Structure of Operation RestartService

7.1.27.1 Request

DeviceManagementService.RestartServiceRequestStructure			*+Structure*	**Request structure which enables to restart one specific (in the ServiceSpecification specified) service.**
	ServiceSpecification	**1:1**	*+ServiceSpecification*	Reference at the restarted services (cf. VDV 301-2-1)

Table 27 Description of DeviceManagementService.RestartServiceRequestStructure

7.1.27.2 Response

For the acknowledge of the request the DataAcceptedResponseStructure (cf VDV 301-2-1) is used.

7.1.28 Data Structures of Operation GetAllSubdeviceInformation

7.1.28.1 Request

Because of being a **`Get`** operation, there is no request structure for operation ***`GetAllSubdeviceInformation`***.

7.1.28.2 Response

DeviceManagementService. GetAllSubdeviceInformationResponse				*+Structure*	**Response structure of operation *GetAllSubdeviceInformation***
				choice	One of the structures below
	a	***DeviceManagementService.GetAllSubdeviceInformationResponseData***	**–1:1**	*+DeviceManagementService.GetAllSubdeviceInformationResponseData*	Detailed response structure for static parameters of subdevices (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 28 Description of DeviceManagementService.GetAllSubdeviceInformationResponse

DeviceManagementService. GetAllSubdeviceInformationResponseData			*+Structure*	**Detailed response structure for static parameters of subdevices**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	SubdeviceInformation List	**1:***	*+SubdeviceInformation*	List of static parameters of subdevices (cf. table below)

Table 29 Description of DeviceManagementService.GetAllSubdeviceInformationResponseData

DeviceManagementService. SubdeviceInformation			*+Structure*	**Static parameters of subdevices**
	SubdeviceName	**1:1**	*IBIS-IP.string*	Name of subdevice
	DeviceInformation	**1:1**	*+DeviceInformation*	Static device parameters (cf. VDV 301-2-1)

Table 30 Description of DeviceManagementService.SubdeviceInformationStructure

7.1.29 Data Structures of Operation SubscribeAllSubdeviceInformation

Data structures described in document VDV 301-2-1 are used to establish subscription.

7.1.30 Data Structures of Operation UnsubscribeAllSubdeviceInformation

Data structures described in document VDV 301-2-1 are used to terminate subscription.

7.1.31 Data Structures of Operation GetDeviceStatusInformation

7.1.31.1 Request

Because of being a **`Get`** operation, there is no request structure for operation ***`GetDeviceStatusInformation`***.

7.1.31.2 Response

DeviceManagementService. GetDeviceStatusInformationResponse				*+Structure*	**Response structure of operation *GetDeviceStatusInformation***
				choice	One of the structures below
	a	***DeviceManagementService.DeviceStatusInformationResponseData***	**–1:1**	*+DeviceManagementService.GetDeviceStatusInformationResponseData*	Detailed response structure for detailed device status (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 31 Description of DeviceManagementService.GetDeviceStatusInformationResponse

DeviceManagementService. GetDeviceStatusInformationResponseData			*+Structure*	**Detailed response structure for detailed device status**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	DeviceStatusInformation	**1:1**	*+DeviceStatusInformation*	Detailed device status (cf. table below)

Table 32 Description of DeviceManagementService.GetDeviceStatusInformationResponseData

DeviceManagementService. DeviceStatusInformation			*+Structure*	**Detailed device status**
	DeviceState	**1:1**	*DeviceStateEnumeration*	Common device state (cf. VDV 301-2-1)
	DeviceStatusList	**0:***	*+DeviceStatus*	List of statuses (cf. table below)

Table 33 Description of DeviceManagementService.DeviceStatusInformationStructure

DeviceManagementService.DeviceStatus			*+Structure*	**Status**
	DeviceStatusName	**1:1**	*IBIS-IP.string*	Name of status
	DeviceStatusFlag	**1:1**	*IBIS-IP. boolean*	Activation of status

Table 34 Description of DeviceManagementService.DeviceStatusStructure

7.1.32 Data Structures of Operation SubscribeDeviceStatusInformation

Data structures described in document VDV 301-2-1 are used to establish subscription.

7.1.33 Data Structures of Operation UnsubscribeDeviceStatusInformation

Data structures described in document VDV 301-2-1 are used to terminate subscription.

7.1.34 Data Structures of Operation GetAllSubdeviceStatusInformation

7.1.34.1 Request

Because of being a **`Get`** operation, there is no request structure for operation ***`GetAllSubdeviceStatusInformation`***.

7.1.34.2 Response

DeviceManagementService. GetAllSubdeviceStatusInformation Response				*+Structure*	**Response structure of operation *GetAllSubdeviceStatusInformation***
				choice	One of the structures below
	a	***DeviceManagement Service.GetAllSubdeviceStatusInformationResponseData***	**–1:1**	*+DeviceManagementService.GetAllSubdeviceStatusInformationResponseData*	Detailed response structure for detailed statuses of subdevices (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 35 Description of DeviceManagementService.GetAllSubdeviceStatusInformationResponse

DeviceManagementService. GetAllSubdeviceStatusInformation ResponseData			*+Structure*	**Detailed response structure for detailed statuses of subdevices**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	SubdeviceStatusInformationList	**1:***	*+SubdeviceStatusInformation*	List of detailed statuses of subdevices (cf. table below)

Table 36 Description of DeviceManagementService.GetAllSubdeviceStatusInformationResponseData

DeviceManagementService. SubdeviceStatusInformation			*+Structure*	**Detailed status of subdevice**
	SubdeviceName	**1:1**	*IBIS-IP.string*	Name of subdevice
	DeviceStatusInformation	**1:1**	*+DeviceStatus Information*	Detailed device status (cf. table 27)

Table 37 Description of DeviceManagementService.SubdeviceStatusInformationStructure

7.1.35 Data Structures of Operation SubscribeAllSubdeviceStatusInformation

Data structures described in document VDV 301-2-1 are used to establish subscription.

7.1.36 Data Structures of Operation UnsubscribeAllSubdeviceStatusInformation

Data structures described in document VDV 301-2-1 are used to terminate subscription.

7.1.37 Data Structures of Operation GetAllSubdeviceErrorMessages

7.1.37.1 Request

Because of being a **`Get`** operation, there is no request structure for operation ***`GetAllSubdeviceErrorMessages`***.

7.1.37.2 Response

DeviceManagementService. GetAllSubdeviceErrorMessagesResponse				*+Structure*	**Response structure of operation GetAllSubdeviceErrorMessages**
				choice	One of the structures below
	a	***DeviceManagementService.GetAllSubdeviceErrorMessagesResponseData***	**–1:1**	*+DeviceManagementService.GetAllSubdeviceErrorMessagesResponseData*	Detailed response structure for error messages of subdevices (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 38 Description of DeviceManagementService.GetAllSubdeviceErrorMessagesResponse

DeviceManagementService. GetAllSubdeviceErrorMessages ResponseData			*+Structure*	**Detailed response structure for error messages of subdevices**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	SubdeviceErrorMessagesList	**1:***	*+SubdeviceErrorMessages*	List of error messages of subdevices (cf. table below)

Tabelle 39 Description of DeviceManagementService.GetAllSubdeviceErrorMessagesResponseData

DeviceManagementService. SubdeviceErrorMessages			*+Structure*	**Error messages of subdevice**
	SubdeviceName	**1:1**	*IBIS-IP.string*	Name of subdevice
	ErrorMessage	**10:***	*+Message*	Error messages and warnings only (no status information, cf. VDV 301-2-1)

Table 40 Description of DeviceManagementService.SubdeviceErrorMessagesStructure

7.1.38 Data Structures of Operation SubscribeAllSubdeviceErrorMessages

Data structures described in document VDV 301-2-1 are used to establish subscription.

7.1.39 Data Structures of Operation UnsubscribeAllSubdeviceErrorMessages

Data structures described in document VDV 301-2-1 are used to terminate subscription.

7.1.40 Data Structures of Operation InstallUpdate

This operation asks a peripheral device to install a new update. In case there are other necessary dependencies or update conditions these must be included in the update package itself. There must be only one active update per device at any time. If a device receives a new *InstallUpdate* request while a previous update job is not finished yet the new request must not be accepted.

7.1.40.1 Request

DeviceManagementService. InstallUpdateRequest			*+Structure*	**Request structure of DeviceManagementService**
	UpdateID	**1:1**	*IBIS-IP. NMTOKEN*	Unique id generated by the controller to identify an update job
	UpdateTimestamp	**1:1**	*IBIS-IP. dateTime*	Timestamp used for GetUpdateStates and RetrieveUpdateState responses and for logging
	UpdateURL	**1:1**	*IBIS-IP.anyURI*	URL from which the device shall download the update file
	UpdateFileChecksum	**0:1**	*Checksum Structure*	Optional checksum of update file
	UpdateFileSize	**0:1**	*IBIS-IP. unsignedLong*	Optional size of update file

Table 41 Description of DeviceManagementService.InstallUpdateRequestStructure

The UpdateURL shall be valid until the update process is finished. End of update is signalled by operation FinalizeUpdate.

Minimum requirement for any device is to enable transfer of update file using HTTP.

7.1.40.2 Response

DeviceManagementService. InstallUpdateResponse				*+Structure*	**Response structure of DeviceManagementService**
				choice	One of the structures below
	a	***UpdateAccept***	**–1:1**	*UpdateAcceptEnumeration*	Enumeration (cf. Table 43 below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 42 Description of DeviceManagementService.InstallUpdateResponseStructure

Enumeration	**Defined Values**	**Description**
UpdateAcceptEnumeration	UpdateAccepted URLTypeUnknown NoUpdatesAllowed	Update will be performed URL type has been rejected, e.g. FTP may not supported by some devices

Enumeration	Defined Values	Description
	ToBePostponed	No updates are possible Update has to be postponed

Table 43 Description of UpdateAcceptEnumeration

On accepting an update the device may change to a special update mode. Some desired device functionality may not be available during the update. After accepting an update the device shall handle the corresponding *UpdateID* as a known request parameter. The initial update status shall be *UpdateRunning*.

7.1.41 Data Structures of Operation RetrieveUpdateState

This operation is used to retrieve the update state of active update job. The job is specified by an *UpdateID*.

7.1.41.1 Request

DeviceManagementService. RetrieveUpdateStateRequest			*+Structure*	**Request structure of DeviceManagementService**
	UpdateID	**1:1**	*IBIS-IP. NMTOKEN*	Unique id generated by the controller to identify an update job

Table 44 Description of DeviceManagementService.RetrieveUpdateStateRequestStructure

7.1.41.2 Response

If the requested UpdateID is unknown, an operation error message is returned.

DeviceManagementService. RetrieveUpdateStateResponse				*+Structure*	**Response structure of DeviceManagementService**
				Choice	One of the structures below
	a	***UpdateStateData***	**–1:1**	*+DeviceManagementService. UpdateStateData*	Detailed response structure (cf. Table 46 below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 45 Description of DeviceManagementService.RetrieveUpdateStateResponseStructure

DeviceManagementService. UpdateStateData			*+Structure*	**Response structure of DeviceManagementService**
	UpdateID	**1:1**	*IBIS-IP. NMTOKEN*	Unique id generated by the controller to identify an update job
	UpdateTimestamp	**1:1**	*IBIS-IP. dateTime*	Timestamp given by operation InstallUpdate
	UpdateStatus	**1:1**	*UpdateStatusEnumeration*	Current status of update (cf. Table 47 below)
	UpdateDetails	**0:1**	*IBIS-IP.string*	Optional device specific update log

Table 46 Description of DeviceManagementService.UpdateStateDataStructure

Enumeration	**Defined Values**	**Description**
UpdateStatusEnumeration	UpdateRunning	Update in progress
	DeviceRestartRequired	Device has to be restarted by operation RestartDevice
	DownloadUpdateFileFailed	Specified update file could not be downloaded from URL
	UpdateFileCorrupted	Specified update file is not usable
	UpdateNotNecessary	State of device firmware already as required
	InstallationFailed	Update failed e.g. due to memory write error
	InstallationSuccessful	Update successfully completed

Table 47 Description of UpdateStatusEnumeration

7.1.42 Data Structures of Operation GetUpdateHistory

This operation is used to get the update history of a device. It needs no request data.

7.1.42.1 Request

Because of being a **`Get`** operation, there is no request structure for operation ***`GetUpdateHistory`***.

7.1.42.2 Response

DeviceManagementService. GetUpdateHistoryResponse				*+Structure*	**Response structure of DeviceManagementService**
				Choice	One of the structures below
	a	***UpdateHistory***	**–1:1**	*+DeviceManagementService. UpdateHistory*	Detailed response structure (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 48 Description of DeviceManagementService.GetUpdateHistoryResponseStructure

DeviceManagementService. UpdateHistory			*+Structure*	**Response structure of DeviceManagementService**
	UpdateHistoryEntry	**0:***	*+DeviceManagementService.UpdateHistoryEntry*	List of updates (empty if the device was not updated yet) Cf. table below

Table 49 Description of DeviceManagementService.UpdateHistoryStructure

Minimum requirement for any device allowing updates is an update history depth of 1, i.e. history shall contain at least the last update performed (regardless of its result), if there was any.

DeviceManagementService. UpdateHistoryEntry			*+Structure*	**Response structure of DeviceManagementService**
	UpdateID	**1:1**	*IBIS-IP. NMTOKEN*	Unique id generated by the controller to identify an update job
	UpdateTimestamp	**1:1**	*IBIS-IP. dateTime*	Timestamp given by operation InstallUpdate
	UpdateURL	**1:1**	*IBIS-IP.anyURI*	URL from which the device downloaded the update file
	UpdateStatus	**1:1**	*UpdateStatusEnumeration*	Status of update (cf. Table 47 above) Typically final status (InstallationSuccessfull or InstallationFailed)
	DataVersionList	**0:1**	*+DataVersionList*	Optional list of updated components
	UpdateDetails	**0:1**	*IBIS-IP.string*	Optional device specific update log

Table 50 Description of DeviceManagementService.UpdateHistoryEntryStructure

7.1.43 Data Structures of Operation FinalizeUpdate

Any accepted update job has to be finished by performing the operation *FinalizeUpdate*. The job is specified by an *UpdateID*.

7.1.43.1 Request

DeviceManagementService. FinalizeUpdateRequest			*+Structure*	**Request structure of DeviceManagementService**
	UpdateID	**1:1**	*IBIS-IP. NMTOKEN*	Unique id generated by the controller to identify an update job

Table 51 Description of DeviceManagementService.FinalizeUpdateRequestStructure

7.1.43.2 Response

If the requested *UpdateID* is unknown an operation error message is returned.

DeviceManagementService. FinalizeUpdateResponse				*+Structure*	**Response structure of DeviceManagementService**
				Choice	One of the structures below
	a	***UpdateStatus***	**–1:1**	*UpdateStatus Enumeration*	Status of update on its finalization
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message indicating cause of failure of operation

Table 52 Description of DeviceManagementService.FinalizeUpdateResponseStructure

If the device receives a valid *FinalizeUpdateRequest* it returns to normal operation and the requested *UpdateID* is no longer valid.

Operation *FinalizeUpdate* has to be done in these cases:

1. Device returns any other update status than *UpdateRunning* or *DeviceRestartRequired*.
2. Update has to be cancelled because the update controller is shut down.

7.1.44 Data Structures of Operation FinalizeAllPendingUpdates

All pending update jobs stored in update history are finalized.

7.1.44.1 Request

There are no data used by the request.

7.1.44.2 Response

Common DataAcceptedResponseStructure is used.

Operation FinalizeAllPendingUpdates has to be done if operation InstallUpdate returns NoUpdatesAllowed.

7.2 Dienst SystemDocumentationService

Der Dienst `SystemDocumenationService` ist in IBIS-IP die Umsetzung der Aufgaben der Fachkomponente System-Dokumentation. Seine Hauptaufgaben bestehen darin, Systemmeldungen zu dokumentieren und die Systemkonfiguration zur Verfügung zu stellen. Da das Hauptaugenmerk auf einer Sicherstellung der Informationsübertragung liegt, ist dieser Dienst in IBIS-IP als HTTP-Dienst umgesetzt worden.

Dieser Dienst wird auf dem Gerät, auf dem per Gerätekonfiguration festgelegt ist, dass er gestartet werden soll, mit dem Ausführen der Applikation auf dem Gerät automatisch gestartet.

❖ SystemDocumentationService

The `SystemDocumenationService` is the implementation of the tasks of the System Documentation component with in IBIS-IP.

Its main tasks are to document system messages and make the system configuration available. Since the main focus is on ensuring information transmission, this service has been implemented in IBIS-IP as an HTTP service.

This service is automatically started on the device on which the device configuration specifies that it is to be started when the application is executed on the device.

7.2.1 Operationen SystemDocumentationService

Operation	**Request/ Response**	**Verwendeter Datentyp, Datenstruktur / Used Data type, Data structure**
GetSystemConfiguration	Req.	-
	Resp.	SystemDocumentationService. GetSystemConfigurationResponseStructure
SubscribeSystemConfiguration	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeSystemConfiguration	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
StoreSystemConfiguration	Req.	SystemDocumentationService. StoreSystemConfigurationRequestStructure
	Resp.	DataAcceptedResponseStructure
StoreLogMessages	Req.	SystemDocumentationService. StoreLogMessagesRequestStructure
	Resp.	DataAcceptedResponseStructure
RetrieveLogMessages	Req.	SystemDocumentationService. RetrieveLogMessagesRequestStructure
	Resp.	SystemDocumentationService. RetrieveLogMessagesResponseStructure

Table 53: Description of Operations at the SystemDocumentationService

7.2.2 Data Structure of Operation GetSystemConfiguration

7.2.2.1 Request

Because of being a ***GetsystemConfiguration*** operation, there is no request structure for this operation.

7.2.2.2 Response

SystemDocumentationService.GetSystemConfigurationResponse				*+Structure*	**Response structure with system configuration data**
				Choice	One of the structures below
	a	***SystemConfiguration***	**–1:1**	*+SystemDocumentationService.SystemConfigurationData*	detailed response structure (cf. below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 54: Description of SystemDocumentationService.GetSystemConfigurationResponse

SystemDocumentationService.SystemConfigurationData			*+Structure*	**Response structure with the system configuration**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	ServiceStartList	**1:1**	*+ServiceStartList*	List with information for all needed services (cf. VDV 301-2-1)
	DeviceSpecificationList	**1:1**	*+DeviceSpecificationList*	List with information about all available devices (cf. VDV 301-2-1)
	HertbeatIntervall	0:1	*xs:duration*	Heartbeat value for the cyclic request of the SystemManagementService

Table 55: Description of SystemDocumentationService.SystemConfigurationData

7.2.3 Data Structure of Operation SubscribeSystemConfiguration

For this subscription the data structures from VDV 301-2-1 are used.

7.2.4 Data Structure of Operation UnsubscribeSystemConfiguration

To terminate this subscription the structures of VDV 301-2-1 are required.

7.2.5 Data Structure of Operation StoreSystemConfiguration

7.2.5.1 Request

SystemDocumentationService.StoreSystemConfigurationRequest			*+Structure*	**Request structure to store a new system configuration**
	ServiceStartList	**1:1**	*+ServiceStartList*	List of services to be started (cf. VDV 301-2-1)
	DeviceSpecificationList	**1:1**	*+DeviceSpecificationList*	List of available devices (cf. VDV 301-2-1)
	HertbeatIntervall	0:1	*xs:duration*	Heartbeat value for the cyclic request of the SystemManagementService

Table 56: Description of SystemDocumentationService.StoreSystemConfigurationRequest

7.2.5.2 Response

Since this is a transmission of information to a service, the answer is given with the general structure intended for this purpose (see VDV 301-2-1).

7.2.6 Data Structure of Operation StoreLogMessages

7.2.6.1 Request

SystemDocumentationService.StoreLogMessages Request			*+Structure*	**Request structure for transmission of logging messages**
	LogMessage	0:*	*+LogMessage*	Logging message to be stored (cf. VDV 301-2-1)
	DataVersion	0:*	*+DataVersion*	Information about the data version to be stored (cf. VDV 301-2-1)

Table 57: Description of SystemDocumentationService.StoreLogMessagesRequest

7.2.6.2 Response

Since this is a transmission of information to a service, the answer is given with the general structure intended for this purpose (see VDV 301-2-1).

7.2.7 Data Structure of Operation RetrieveLogMessages

7.2.7.1 Request

SystemDocumentationService.RetrieveLogMessagesRequest			*+Structure*	**Request structure for logging messages**
	InformationType	**1:1**	*SystemDocumentationInformationEnumeration*	Information about the logging message type (cf. table below)
	NumberOfEntries	**1:1**	*xs:int*	Number of entities

Table 58: Description of SystemDocumentationService.RetrieveLogMessagesRequest

7.2.7.2 Response

SystemDocumentationService.RetrieveLogMessagesResponse				*+Structure*	**Response structure with logging messages data**
				choice	One of the structures below
	a	***LogMessageData***	**–1:1**	*+SystemDocumentationService.LogMessageData*	detailed response structure (cf. VDV 301-2-1)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 59: Description of SystemDocumentationService.RetrieveLogMessagesResponse

SystemDocumentationService.LogMessageData			*+Structure*	**Detailed response structure for logging messages**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	Message	**1:***	*+Message*	Message content (cf. VDV 301-2-1)

Table 60: Description of SystemDocumentationService.LogMessageData

7.3 Dienst SystemManagementService

Der Dienst `SystemManagementService` ist in IBIS-IP die Umsetzung der Aufgaben der Fachkomponente System-Management. Seine Hauptaufgaben bestehen darin,

— alle Dienste, die laut Systemkonfiguration zum IBIS-IP-System gehören sollen, zu starten (vgl. Kapitel ? bis 5.3)

— den Status aller Geräte und Dienste des IBIS-IP-Systems zu überwachen

— den Status aller Geräte und Dienste des IBIS-IP-Systems zu veröffentlichen

Dieser Dienst wird auf dem Gerät, auf dem per Gerätekonfiguration festgelegt ist, dass er gestartet werden soll, mit dem Ausführen der Applikation auf dem Gerät automatisch gestartet.

In einem IBIS-IP-System darf es grundsätzlich nur einen `SystemManagementService` geben. Dieser `SystemManagementService` darf sich mit nur einer IBIS-IP-Version eines `SystemDocumentationService` verbinden.

Der Parallelbetrieb zweier `SystemManagementServices` (z. B. mit zwei verschiedenen IBIS-IP-Versionen) entspräche dem Parallelbetrieb zweier IBIS-IP-Systeme. Ein solcher Zustand ist nicht zulässig.

Da das Hauptaugenmerk auf einer Sicherstellung der Informationsübertragung liegt, ist dieser Dienst in IBIS-IP als HTTP-Dienst umgesetzt worden.

❖ SystemManagementService

The `SystemManagementService` in IBIS-IP is the implementation of the tasks of the System Management component. His main tasks are:

— to start all services which, according to the system configuration, should belong to the IBIS IP system (cf. chapter 5.2 to 5.3)

— monitor the status of all devices and services of the IBIS IP system

— publish the status of all devices and services of the IBIS IP system

This service is automatically started on the device on which the device configuration specifies that it is to be started when the application is executed on the device.

In an IBIS IP system, there may only be one `SystemManagementService` in principle. This `SystemManagementService` may only connect to one IBIS-IP version of a `SystemDocumentationService`.

The parallel operation of two `SystemManagementServices` (e. g. with two different IBIS-IP versions) would correspond to the parallel operation of two IBIS-IP systems. Such a condition is not permitted.

Since the main focus is on ensuring information transmission, this service has been implemented in IBIS-IP as an HTTP service.

7.3.1 Operationen SystemManagementService

Operation	**Request/ Response**	**Verwendeter Datentyp, Datenstruktur / Used data types, Data structures**
GetDeviceStatus	Req.	-
	Resp.	SystemManagementService. GetDeviceStatusResponseStructure
SubscribeDeviceStatus	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeDeviceStatus	Req.	UnsubscribeRequestStructure
	Resp.	UnsubscribeResponseStructure
GetServiceStatus	Req.	-
	Resp.	SystemManagementService. GetServiceStatusResponseStructure
SubscribeServiceStatus	Req.	SubscribeRequestStructure
	Resp.	SubscribeResponseStructure
UnsubscribeServiceStatus		UnsubscribeRequestStructure
		UnsubscribeResponseStructure

Tabelle 61 Beschreibung von Operationen des SystemManagementService / Description of Operations at the SystemManagementService

7.3.2 Data Structure of Operation GetDeviceStatus

7.3.2.1 Request

Because of being a ***GetDeviceStatus*** operation, there is no request structure for this operation.

7.3.2.2 Response

SystemManagementService.GetDeviceStatusResponse				*+Structure*	**Response structure for the device state of all devices**
				choice	One of the choices below
	a	***SystemManagementService.GetDeviceStatusResponseData***	**–1:1**	*+ SystemManagementService.GetDeviceStatusResponseData*	Detailed response structure with information about the status of all devices (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 62: Description of SystemManagementService.GetDeviceStatusResponse

SystemManagementService.GetDeviceStatusResponseData			*+Structure*	**Detailed response structure with information about the status of all devices**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	DeviceSpecificationWithStateList	**1:1**	*+DeviceSpecificationWithStateList*	Device list with all device states (vgl. VDV 301-2-1)

Table 63: Description of SystemManagementService.GetDeviceStatusResponseData

7.3.3 Data Structure of Operation SubscribeDeviceStatus

For this subscription the data structures from chapters VDV 301-2-1 are used.

7.3.4 Data Structure of Operation UnsubscribeDeviceStatus

To terminate this subscription the structures of chapters VDV 301-2-1 are used.

7.3.5 Data Structure of Operation GetSystemStatus

7.3.5.1 Request

Because of being a ***`GetServiceStatus`*** operation, there is no request structure for this operation.

7.3.5.2 Response

SystemManagementService.GetServiceStatusResponse				*+Structure*	**Response structure with the service state of all devices in the system**
				choice	One of the choices below
	a	***SystemManagementService.GetServiceStatusResponseData***	**–1:1**	*+SystemManagementService.GetServiceStatusResponseData*	Detailed response structure with the device state of all services in the system (cf. table below)
	b	***OperationErrorMessage***		*IBIS-IP.string*	Error message

Table 64: Description of SystemManagementService.GetServiceStatusResponse

SystemManagementService.GetServiceStatusResponseData			*+Structure*	**Detailed response structure with the device state of all services in the system**
	TimeStamp	**1:1**	*IBIS-IP.dateTime*	Response time stamp
	ServiceIdentificationWithStateList	**1:1**	*+ServiceIdentificationWithStateList*	Service specification and state list (cf. VDV 301-2-1)

Table 65: Description of SystemManagementService.GetServiceStatusResponseData

7.3.6 Data Structure of Operation SubscribeSystemStatus

For this subscription the data structures from chapters VDV 301-2-1 are used.

7.3.7 Data Structure of Operation UnsubscribeSystemStatus

To terminate this subscription the structures of chapters VDV 301-2-1 are used.

8 Versionshistorie / Version History

8.1 Version 2.0

8.1.1 Funktionale Erweiterungen
Functional Upgrade

- Keine/none

8.1.2 Technische Ergänzungen/Korrekturen
Technical Upgrade/Corrections

- *SystemDocumentationService.*
  SystemConfigurationData.HeartbeatIntervall
  StoreSystemConfigurationRequestStructure.HeartbeatIntervall
  Schreibfehler korrigiert nach *HeartbeatInterval* (vgl. 7.2.2.2)
  typo corrected to *HeartbeatInterval* (cf. 7.2.2.2)

- Fehlenden Suffix „_tcp.“ oder „_udp.“ in den Beispielen zum SRV-Record-Eintrag „Proto“ des DNS-SD ergänzt (vgl. 3.3.1)
  Missing suffix „_tcp.“ or „_udp.“ added in examples for SRV record entry „Proto“of DNS-SD (cf. 3.3.1)

- Überflüssigen Punkt aus dem Beispiel zum SRV-Record-Eintrag „Name“ des DNS-SD entfernt (vgl. 3.3.1)
  Needless dot removed from example for SRV record entry „Name“of DNS-SD (cf. 3.3.1)

- Überflüssigen Punkt aus dem Beispiel zum SRV-Record-Eintrag „Target“ des DNS-SD entfernt (vgl. 3.3.1)
  Needless dot removed from example for SRV record entry „Target“of DNS-SD (cf. 3.3.1)

- Eindeutiger Service-Name im SRV-Record des DNS-SD (vgl. 3.3.1)
  unique service name in SRV record of DNS-SD (cf. 3.3.1)

8.1.3 Textliche Korrekturen
Textual Corrections

- Beschreibung der gemeinsamen Datenstrukturen und Aufzählungstypen in das neue Dokument VDV-301-2-1 verschoben
  Description of common data structures and enumerations moved to new document VDV-301-2-1

- Best-Practice-Hinweise in den Abschnitten 2.1.1 und 5.2 ergänzt
  Best practice hints in sections 2.1.1 and 5.2 added

- Best-Practice-Hinweis und Beispiel im Abschnitt 3.3.1 ergänzt
  Best practice hint and example in section 3.3.1 added

- Autorenliste vervollständigt

8.2 Version 2.1

8.2.1 Funktionale Erweiterungen
Functional Upgrade

- DeviceManagementService erweitert um Operationen zur Abfrage detaillierter Informationen zum Gerätestatus inklusive der Unterstützung von eigenständigen Gerätekomponenten
  DeviceManagementService extended by operations to query detailed information on device status including support of subdevices

- DeviceManagementService erweitert um Operationen zur Realisierung des Update-Verfahrens
  DeviceManagementService extended by operations to realize update procedure

8.2.2 Technische Ergänzungen/Korrekturen
Technical Upgrade/Corrections

- Keine/none

9 Begriffe

Die bereits im Teil 1 definierten Begriffe werden an dieser Stelle nicht wiederholt.

Abkürzung	Beschreibung
Abonnement	Mit einem solchen Verfahren wird der anfragende Teilnehmer automatisch mit aktuellen Informationen versorgt.
AnnouncementSystem	Beschreibt die Geräteklasse Elektroakustische-Anlage inkl. Fahrgastsprechstellen.
Applikation	Als Applikation wird die auf den Geräten laufende Software bezeichnet. Diese herstellerspezifische Software bedient allerdings spezifizierte Schnittstellen.
BeaconLocationService	Ein IBIS-IP Dienst der die Informationen der Ortsbaken übermittelt.
CustomerInformationService	Als zentrale Informationsstelle für alle Belange der Fahrgastinformation sorgt in IBIS-IP dieser Dienst für eine konsistente Bereitstellung aller Daten.
DeviceManagementService	Der auf jedem Gerät in IBIS-IP vorherrschender Dienst stellt Informationen über das Gerät und die laufenden Dienste bereit.
DistanceLocationService	Die Auswertung der Odometerimpulse aus der Wegstrecke übernimmt in IBIS-IP dieser Dienst.
FrontDisplay	Entspricht der Geräteklasse Fahrzeugzielanzeige.
Geräteklassen	Zur Transparenz der angeschlossenen Geräte im Netzwerk wurden Geräteklassen spezifiziert, die alle derzeit denkbaren Geräte abdecken.
GNSS-Koordinaten	Auf Grund einer satellitenbasierte Ortung (z. B. GPS, Galileo) gewonnene Koordinaten eines Punktes.
GNSSLocationService	Für die Verteilung von in Koordinaten beschreibbaren aktuellen Standort des Fahrzeugs auf Basis des NMEA Telegramms.
HTTP-GET	Es handelt sich um eine Anfragefunktion des HTTP. I. d. R. werden dieser Frage keine Daten mit gesendet.
HTTP-POST	Es handelt sich um eine Anfragefunktion des HTTP. I. d. R. werden dieser Frage Daten mit gesendet.
HTTP-Protokoll	Mit dieser Bezeichnung wird innerhalb von IBIS-IP eine Übertragungsart für sich eher selten ändernde Informationen verwendet.
HTTP-Protokoll-Stack	Bezeichnet die komplette Protokolleinheit (von TCP bis HTTP) die für eine sichere Übertragung notwendig ist.
InteriorDisplay	Beschreibt die Geräteklasse der Anzeiger im Innenraum des Fahrzeuges, des können Fahrgastinformationsanzeiger aber auch digitale Werbeanzeiger sein.
JourneyInformationService	Dieser Dienst bietet Fahrplandaten den abfragenden Diensten zur Verfügung.
JourneyWeb	Ist ein XML-Protokoll für Fahrgastinformation.
MobileInterface	Beschreibt die Geräteklasse der Schnittstelle an der die Fahrgäste mit ihrem mobilen Endgerät Informationen vom Fahrzeug abholen können.
Multicast	Kommunikationsverfahren zum Ansprechen vieler dedizierter Empfänger.
Multicast-Gruppe	Beschreibt die dedizierten Empfänger einer Multicast-Nachricht.
NetworkLocationService	Der Dienst stellt Informationen über den aktuellen Standort auf einem geplanten Linienfahrweg in einem Netz des ÖV bereit.
Odometer	Auf raddrehzahl basierter Entfernungsmesser, der eine bestimmte Anzahl Impuls pro gefahrenem Meter produziert.

Abkürzung	Beschreibung
OnBordUnit	Entspricht der Geräteklasse des Zentralrechners aus IBIS Zeiten.
Other (Geräteklasse)	Hierunter lassen sich Geräte anbinden die noch nicht in diesem Standarad Klassifiziert wurden.
PassengerCountigService	Dieser Dienst stellt die Fahrgastzähldaten einer jeder Tür dem IBIS-IP-System zur Verfügung.
Port	Ist eine „Zugangstür“ des Protokolls, das der Applikation eine eindeutige Quellenzuordnung ermöglicht
Request/Response	Beschreibt den Kommunikationsablauf indem einer Frage von Teilnehmer A an Teilnehmer B eine Antwort von Teilnehmer B folgt.
SideDisplay	Beschreibt die Geräteklasse der seitliche Aussenanzeige an einem Fahrzeug.
SRV-Records	Gemäß RFC 2782 lassen sich mit diesem Textfeld zusätzliche Informationen (z. B. Rechnername) an den DNS-Dienst anhängen.
SystemDocumentationServic e	Der Dienst ist in IBIS-IP dafür verantwortlich Systemmeldungen zu dokumentieren und die Systemkonfiguration zur Verfügung zu stellen.
SystemManagementService	Die Hauptaufgaben bestehen darin, über den aktuellen Status der Geräte und der Dienste zu informieren, sowie das System zu koordinieren.
TestDevice	Platzhalter für die Geräteklasse der Testgeräte, die im Falle einer Verifikation des Systems angeschlossen werden.
TicketingService	Der Dienst stellt Ticketing Funktionen im IBIS-IP-System zur Verfügung.
TicketVendingMachine	Beschreibt die Geräteklasse der Fahrscheinverkaufsgeräte innerhalb des Fahrzeuges. Dies können fahrerbediente aber auch autarke Automaten sein.
TimeService	Der Zeitdienst stellt via SNTP dem System die aktuelle Uhrzeit, das Datum und die Zeitzone zur Verfügung.
TransModel	Ist ein Referenz Daten Model nach EN12896.
TXT-Records	Gemäß RFC 1464 lassen sich mit diesem Textfeld zusätzliche Informationen (z. B. Rechnername) an den DNS-Dienst anhängen.
UDP-Protokoll	Mit dieser Bezeichnung wird innerhalb von IBIS-IP eine Übertragungsart für sich zyklisch ändernde Informationen verwendet.
UML-Konvention	Formatvorgaben aus der UML-Programmiersprache
USB-Stick	Mobiler Datenspeicher
Validator	Geräteklasse für Stempelautomat oder elektronischer Entwerter für E-Tickets.
VideoSystem	Beschreibt die Geräteklasse des Videoüberwachungssystems, das über IBIS-IP gesteuert wird bzw. Videodaten überträgt
Zeitserver	Ist eine zentrale Komponente innerhalb von IBIS-IP, die die Systemzeit zur Verfügung stellt.
Zero Conf	Ist eine Methode und eine Zusammenfassung spezieller Technologien zur automatischen Koppelung von Geräten in einem IP Netzwerk

Term	Description
AnnouncementSystem	Describes the device class of electro-acoustic systems incl. passenger communication unit
Application	The software on the device is referred to as an application. However, this proprietary software uses specified interfaces..
BeaconLocationService	An IBIS-IP service, which transmits information of the location beacons.
CustomerInformationService	As a central point of information for all aspects of passenger information in IBIS-IP this service is responsible for a consistent delivery of all data.
Device class	For the naming of all connected devices in the network device-classes are specified for all currently known devices
DeviceManagementService	Der auf jedem Gerät in IBIS-IP vorherrschender Dienst stellt Informationen über das Gerät und die laufenden Dienste bereit.
DeviceManagementService	The on every device in IBIS-IP existing service provides information about the device and the running services.
DistanceLocationService	The evaluation of the odometer pulses in IBIS IP is done by this service.
FrontDisplay	Corresponds to the device class FrontDisplay, a display on front of a bus indicating its destination.
GNSS coordinates	Due to a satellite-based positioning (e.g. GPS, Galileo) obtained coordinates of a point.
GNSSLocationService	IBIS-IP service which provides the positioning information of a vehicle on base of NMEA-telegrams
HTTP-GET	Request-Function of HTTP, with this function no data is sent
HTTP-POST	Request-Function of HTTP, with this function data could be sent
HTTP-Protocol	Communication protocol in IBIS-IP which is used for event triggered changing data
HTTP-Protocol-Stack	Term for the protocol unit (with TCP and HTTP) for a secure data transmission
InteriorDisplay	Device class for the description of displays inside vehicles independent of the shown contents.
JourneyInformationService	This service provides scheduling data to requesting services/devices.
MobileInterface	Describes the device class of the interface where the passengers with their mobile device can receive information from the vehicle.
Multicast	Communication method for addressing many dedicated receiver.
Multicast group	Describes the dedicated receiver of a multicast message.
NetworkLocationService	This service provides information of the current position on the planned vehicle journey in the network of public transport.
Odometer	A wheel-based rangefinder which produces a certain number of pulses per driven meter.
OnBoardUnit	Device class for the description of the central unit (refers to the master device in IBIS-systems).
Other (Deviceclass)	With this deviceclass devices which are not classified in this standard could be integrated.
PassengerCountigService	This service provides the data of passenger counting on each door in an IBIS-IP-system.

Term	Description
Port	Is an “access” of the protocol, which allows the application a unique source identification
Request/Response	Describes the communication process where a request of communication partner A is answered by a response of communication partner B
SideDisplay	Describes the device class of the lateral outer display on a vehicle.
SRV-Records	Due to RFC 2782 with this record additional information (e.g. device name) to a DNS registration could be provided.
Subscription	With this method, the requesting party is automatically provided with current information.
SystemDocumentationService	This service in IBIS-IP is responsible for documentation of system messages and provision of the system configuration.
SystemManagementService	The main task of the SystemManagementService in IBIS-IP is to inform about the current state of devices and services and to coordinate the system.
TestDevice	Placeholder for the device class of test devices which are connected in case of verification of the system.
TicketingService	This service provides ticketing functions in the IBIS-IP system.
TicketVendingMachine	Deviceclass which describes vending machines for tickets onboard vehicles. They could be managed by the driver as well be autonomous.
Timeserver	Is a key component within IBIS-IP, which provides the system time
TimeService	The TimeService provides via SNTP the current time, date, and time zone.
TransModel	Reference data model based on EN12896.
TXT-Records	According to RFC 1464 this could be used for adding additional information (e.g. devicename) to a DNS-service.
UDP-Protocol	IP-Protocol which in IBIS-IP is used fort he transmission of cyclic changing information.
UML-Convention	Format specifications from UML programming language
USB-Stick	Mobile Data Storage Device
Validator	Deviceclass which is used for stamping validators as well as for the electronic validation of E-tickets.
VideoSystem	Deviceclass which is used for configuration and data transmission of CCTV-systems
Zero Conf	Method and Summary of special technologies for automated identification and configuration of devices in an IP-network

Regelwerke – Normen und Empfehlungen

(1)	CEN/TS 13149-7	Öffentlicher Verkehr - Planungs- und Steuerungssysteme für Straßenfahrzeuge - Teil 7: IP-basierende Vernetzung in einem Fahrzeug, Netzwerk- und Systemarchitektur (FprCEN/TS 13149-7:2015)
(2)	CEN/TS 13149-8	Öffentlicher Verkehr - Planungs- und Steuerungssysteme für Straßenfahrzeuge - Teil 8: Physikalische Schicht für IP-Kommunikation; Englische Fassung CEN/TS 13149-8:2013
(3)	EN 15531	Service Interface for Real Time Information
(4)	EN 1545	Identification card systems.
(5)	VDV 301-1	Internetprotokoll basiertes integriertes Bordinformationssystem IBIS-IP - Teil 1: Systemarchitektur
(6)	VDV 301-2	Internetprotokoll basiertes integriertes Bordinformationssystem IBIS-IP - Teil 2: Schnitstellenspezifikation
(7)	VDV 301-2-1	IBIS-IP Beschreibung der Dienste, Gemeinsame Datenstrukturen und Aufzählungstypen
(8)	VDV 300	Integriertes Bordinformationssystem IBIS
(9)	VDV 3001	Kommunikation im ÖV (IP-KOM-ÖV) - Technische Anforderungen für Anwendungen im Integrierten Bordinformationssystem (IBIS)
(10)	VDV 453	VDV-Ist-Datenschnittstellen
(11)	VDV 454	VDV-Ist-Datenschnittstellen
(12)	Bibliotheken „Bonjour“, „Avahi“	Für Windows- und MAC-OS-Betriebssysteme gibt es die Bibliotheken von Bonjour, unter Linux die Bibliotheken von Avahi, beide Bibliotheken stehen im Quelltext unter einer Opensource-Lizenz zur Verfügung.
(13)	RFC 3927	Dynamic Configuration of IPv4 Link-Local Addresses
(14)	RFC 6365	Terminology Used in Internationalization in the IETF
(15)	RFC 2782	A DNS RR for specifying the location of services (DNS SRV)
(16)	RFC 1035	Domain Names – Iimplementation an Specification
(17)	RFC 1464	Using the Domain Name System to Store Arbitrary String Attributes
(18)	RFC 4330	Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI

Die IBIS-IP XSD-Dateien stehen unter www.vdv.de/ip-kom-oev.aspx zum Download bereit.

Bildverzeichnis

Tabellenverzeichnis

Impressum

Verband Deutscher Verkehrsunternehmen e. V. (VDV)
Kamekestraße 37-39 · 50672 Köln
T 0221 57979-0 · F 0221 57979-8000
info@vdv.de · www.vdv.de

Ansprechpartner

Dipl.-Ing. Berthold Radermacher
T 0221 57979-141
F 0221 57979-8141
radermacher@vdv.de

Verband Deutscher Verkehrsunternehmen e. V. (VDV)
Kamekestraße 37-39 · 50672 Köln
T 0221 57979-0 · F 0221 57979-8000
info@vdv.de · www.vdv.de