VDV Die Verkehrsunternehmen

VDV-Schrift 301-2

02/2021

IBIS-IP Beschreibung der Dienste / Service description

Allgemeine Konventionen / General conventions V2.3

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.

VDV-Schrift 301-2 11/2020

IBIS-IP Beschreibung der Dienste / Service description

Allgemeine Konventionen / General conventions V2.3

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

Der Anwender ist für die sorgfältige und ordnungsgemäße Anwendung der Schrift verantwortlich. Stellt der Anwender Gefährdungen oder Unregelmäßigkeiten im Zusammenhang mit der Anwendung dieser Schrift fest, wird eine unmittelbare Benachrichtigung an den VDV erbeten. Eine Haftung des VDV oder der Mitwirkenden an der Schrift ist, soweit gesetzlich zulässig, ausgeschlossen.

© Verband Deutscher Verkehrsunternehmen e. V. Köln 2015 | Alle Rechte, einschließlich des Nachdrucks von Auszügen, der fotomechanischen oder datenverarbeitungstechnischen Wiedergabe und der Übersetzung, vorbehalten.

Vorwort

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 beschrieben.

Foreword

The further development of VDV-301 has shown that the integration of all services in a single document 301-2 Version 1.0 is cumbersome to use. Therefore, the individual services were separated into individual fonts.

The subdivision of the services into individual documents is based on the simpler adaptation and release. Currently, many manufacturers are in the process of implementing the standard. Therefore, there is a greater need for adaptation. New services can also be adapted independently of the existing ones.

The VDV-301 documents consist of basic documents such as the VDV-301-1 and VDV-301-2 as well as the individual descriptions of the services (VDV-301-2-x).

VDV-301-2 describes the technical basics.

Inhaltsverzeichnis

Vorwort 4
Foreword 4
Inhaltsverzeichnis 5
Abkürzungen /Abbreviation 8
1 Einführung in IBIS-IP 9
1.1 Anforderungen an Geräte in IBIS-IP 9
1.2 Einbaukennung 9
1.3 Geräteklassen 9
1.4 Notationen 11
1.5 Versionierung in IBIS-IP 11
1.6 Dokumentationen der Dienste 12
1 Introduction to IBIS-IP 13
1.1 Requirements on Devices in IBIS-IP 13
1.2 Installation Identifier 13
1.3 Device Classes 13
1.4 Notations 15
1.5 IBIS-IP- Version 15
1.6 Documentation of services 16
2 Verwendete Kommunikationsprotokolle 17
2.1 Adressierung 17
2.1.1 IP-Adressen 17
2.1.1 Subnetzmasken/Gateways 17
2.2 Konfigurationsparameter für TCP und UDP 17
2.3 Gültigkeitsdauer von Informationen 17
2.3.1 Zyklische Informationen 18
2.3.2 Ereignisgesteuerte Informationen 18
2.4 Verwendung des UDP- und HTTP-Protokolls 18
2.5 Weitere Protokolle 19
2 Communication Protocols used 20
2.1 Adressing 20
2.1.1 IP Addresses 20
2.1.2 Subnet Masks/Gateways 20
2.2 Configuration Parameters for TCP and UDP 20
2.3 Information Validity Period 20
2.3.1 Periodic Information 21
2.3.2 Event triggered Information 21
2.4 Use of the UDP and HTTP Protocols 21
2.5 Other Protocols 22
3 Veröffentlichung und Kommunikation der Dienste 23
3.1 Von Fachkomponenten zu Diensten 23
3.2 Spezifizierte Dienste 25
3.3 Veröffentlichung via DNS-SD 25
3.3.1 Nutzung des SRV-Records 25
3.3.1 Nutzung des TXT-Records 27
3.4 Veröffentlichung von UDP-Diensten 28
3.5 Veröffentlichung von HTTP-Diensten 28
3 Service Publication and Communication 29
3.1 From functional Components to Services 29
3.2 Specified Services 31

3.3 Publication via DNS-SD ... 31
3.3.1 Use of SRV Records ... 31
3.3.2 Use of TXT Records ... 33
3.4 Publication of UDP Services ... 33
3.5 Publication of HTTP Services ... 34
4 Konventionen für Dienste und Datenstrukturen ... 35
4.1 Konventionen für HTTP-Dienste ... 35
4.1.1 Verwendung von HTTP-POST und HTTP-GET ... 35
4.1.1.1 Operationsaufrufe mit HTTP-POST ... 36
4.1.1.2 Operationsaufrufe mit HTTP-GET ... 36
4.1.2 Namenskonventionen für Operationen ... 36
4.1.2.1 Rein datenorientierte Operationen ... 37
4.1.2.1.1 Einmalige Abfrage von aktuell gültigen Daten bzw. Datenstrukturen ... 38
4.1.2.1.2 Abonnieren auf Daten bzw. Datenstrukturen ... 38
4.1.2.1.3 Beenden eines Abonnements auf Daten bzw. Datenstrukturen ... 38
4.1.2.1.4 Operationen zur Abfrage von spezifischen Informationen ... 39
4.1.2.1.5 Operationen zur Abfrage von Datenlisten aus der Grunddatenversorgung ... 39
4.1.2.2 Steuerungsoperationen ... 39
4.1.2.2.1 Neustart auslösen ... 40
4.1.2.3 Operationen zur Gültigkeitsprüfung ... 40
4.1.3 Konvention zu Get/Subscribe/Unsubscribe ... 40
4.1.4 Unterscheidung zwischen Get und Retrieve ... 41
4.2 Konventionen für UDP-Dienste ... 41
4.3 Konventionen für besonderes Dienstverhalten ... 41
4.3.1 Konventionen für Zustände ... 41
4.3.2 Konventionen für Fehlermeldungen ... 42
4.4 Konventionen für die Datenstrukturierung ... 42
4 Conventions for Services and Data Structures ... 44
4.1 Conventions for HTTP Services ... 44
4.1.1 Use of HTTP-POST and HTTP-GET ... 44
4.1.1.1 Operation Calls with HTTP-POST ... 44
4.1.1.2 Operation Calls with HTTP-GET ... 45
4.1.2 Naming Conventions for Operations ... 45
4.1.2.1 Purely data-oriented Operations ... 46
4.1.2.1.1 Single Request of currently valid Data and/or Data Structures ... 46
4.1.2.1.2 Subscription to Data and/or Data Structures ... 46
4.1.2.1.3 Termination of a Subscription to Data and/or Data Structures ... 47
4.1.2.1.4 Operations for requesting specific Information ... 47
4.1.2.1.5 Operations for requesting Data Lists from the basic Data Supply ... 48
4.1.2.2 Control Operations ... 48
4.1.2.2.1 Trigger Restart ... 48
4.1.2.3 Operations for Validity Checking ... 48
4.1.3 Conventions for Get/Subscribe/Unsubscribe ... 48
4.1.4 Differentiation between Get and Retrieve ... 49
4.2 Conventions for UDP Services ... 49
4.3 Conventions for special Service Behavior ... 49
4.3.1 Conventions for States ... 49
4.3.2 Conventions for Error Messages ... 50
4.4 Conventions for Data Structuring ... 50
5 Prinzipien in IBIS-IP ... 52
5.1 Grundlagen ... 52
5.1.1 Nur ein Dienst pro Fachlichkeit ... 52
5.1.2 Fachliche Identifikation von Gerätem ... 52
5.1.3 Technische Identifikation von Geräten ... 53

5.1.4 Fachliche Identifikation von Diensten ... 53
5.1.4.1 Technische Identifikation von Diensten ... 53
5.2 Das Konzept des Systemstarts ... 54
5.2.1 Start der Dienste ... 55
5.2.2 Veröffentlichung der Dienste ... 55
5.2.3 Verbindungsaufbau mit den veröffentlichten Diensten und Ausführung fachlicher Aufgaben ... 56
5 Principles in IBIS-IP ... 57
5.1 Basics ... 57
5.1.1 One Service only per Functionality ... 57
5.1.2 Functional Identification of devices ... 57
5.1.3 Technical Identification of devices ... 58
5.1.4 Functional Identification of services ... 58
5.1.4.1 Technical Identification of Services ... 58
5.2 The System Start Concept ... 59
5.2.1 Start of the Services ... 60
5.2.2 Publication of the Services ... 60
5.2.3 Establish Connection with the published Services and Execution of functional Tasks ... 61
6 Strukturierung der Informationsinhalte ... 62
6.1 Notation der XML-Elemente und -Strukturen ... 62
6.1.1 Darstellung von XML-Elementen im Text ... 62
6.1.2 Tabellennotation für Operationen ... 63
6.1.3 Tabellennotation von XML-Strukturen ... 63
6.1.3.1 Gruppierung ... 64
6.1.3.2 Elementname ... 64
6.1.3.3 Multiplizität & Choice (Min:Max) ... 64
6.1.3.4 Datentyp ... 64
6.1.3.5 Erläuterung ... 65
6 Structuring of the information contents ... 66
6.1 Notation of XML Elements and Structures ... 66
6.1.1 Representation of XML Elements in Text ... 66
6.1.2 Table Notations for Operations ... 67
6.1.3 Table Notation of XML Structures ... 67
6.1.3.1 Grouping ... 68
6.1.3.2 Element Name ... 68
6.1.3.3 Multiplicity & Choice (Min:Max) ... 68
6.1.3.4 Data Type ... 68
6.1.3.5 Explanation ... 69
7 Versionshistorie ... 70
7 Version History ... 70
7.1 Version 2.2 ... 70
7.1.1 Funktionale Erweiterungen ... 70
7.1.1 Functional Upgrade ... 70
7.1.2 Technische Ergänzungen/Korrekturen ... 70
7.1.2 Technical Upgrade/Corrections ... 70
8 Begriffe ... 72
8 Terms ... 74
Regelwerke – Normen und Empfehlungen / Standards and recommendations ... 76
Bildverzeichnis / List of figures ... 77
Tabellenverzeichnis / List of tables ... 78
Impressum ... 79

Abkürzungen /Abbreviation

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.

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.
- Optional: IBIS-IP Geräte können Gerätstatus und Geräteinformationen von angeschlossenen Geräten melden. Diese Geräte werden als Sub-Devices bezeichnet. Sub-Devices führen keine IBIS-IP-Dienste aus und nehmen nicht an der IBIS-IP Kommunikation teil.

1.2 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, IP-Adresse 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).

1.3 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 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.
MobileInterface	Kundenkommunikations - 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.
CombiDevice	Mehrzweck Gerät	Hierunter lassen sich Geräte anbinden, die mehrere Funktionalitäten ausführen können.

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 5.1.2)

1.4 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 /`

1.5 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.

Die IBIS-IP Version besteht aus zwei Ziffern, die durch einen Punkt getrennt sind und folgendermassen formatiert sind (IBIS-IP Version: «Hauptversion ».« Unterversion», Beispiel IBIS-IP Version: ‘2.1‘).

Die IBIS-IP Version wird aus folgenden Zeichen gebildet (‘0‘,‘1‘,‘2‘,‘3‘,‘4‘,‘5‘,‘6‘,‘7‘,‘8‘,‘9‘ or ‘.’). Versionen, die andere Zeichen enthalten oder anders formatiert sind (wie zum Beispiel die Versionsangaben ‘2.1a’, ‘2.0.1‘ oder ’.2.1’), sind ungültig.

Mit diesen Definitionen kann die am besten passende konforme Serviceversion bestimmt und verwendet werden.

Client und Server sind kompatibel, wenn die Versionskennung übereinstimmen; ungültige Versionen werden abgelehnt.

Falls der Server mehrere kompatible Dienste anbietet, verbindet sich der Client mit dem Dienst, der die "höchste" kompatible IBIS-IP Version meldet.

Projektspezifische Erweiterungen oder Abweichungen vom Standard sind möglich. In diesem Fall sind ungültige Versionskennungen (z. B. ‘2.1a‘) zu verwenden und projektspezifisch zu vereinbaren.

1.6 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)

1 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 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).*
— Optional: IBIS-IP devices can report device status and device information from connected devices. These devices are referred to as sub-devices. Sub-Devices do not perform IBIS-IP services and do not participate in IBIS-IP communication.

1.2 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, IP-Address 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).

1.3 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.

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).
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.
CombiDevice	Multi-purpose device	This can be used to connect devices that can perform several functions.

Table 1: 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 5.1.2)

1.4 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 /`

1.5 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.

The IBIS-IP version consists of two digits separated by a period and formatted as follows (IBIS-IP version: "Main version "." Subversion", example IBIS-IP version: '2.1').

The IBIS-IP version consists of the following characters ('0','1','2','2','3','4','5','6','7','8','9' or '.'). Versions that contain other characters or are formatted differently (such as the version information '2.1a', '2.0.1' or '.2.1') are invalid.

With these definitions, the most appropriate compliant service version can be determined and used.

Client and server are compatible if the version identifiers match; invalid versions are rejected.

If the server offers multiple compatible services, the client connects to the service that reports the "highest" compatible IBIS IP version.

Project specific extensions or deviations from the standard are possible. In this case, invalid version identifiers (e.g. '2.1a') must be used and agreed upon for each project.

1.6 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.

2.1 Adressierung

2.1.1 IP-Adressen

Vorgaben for die IP-Adressvergabe existieren nicht. 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.

2.1.1 Subnetzmasken/Gateways

Es ist projektspezifisch festzulegen, welche Adressen verwendet werden sollen.

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).

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.

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.

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.

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.
  IBIS-IP verwendet die etablierte http-Version 1.1.

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.

2 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 Adressing

2.1.1 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 Subnet Masks/Gateways

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

2.2 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 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 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 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 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.
IBIS-IP uses the established http version 1.1.

2.5 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) 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.

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:

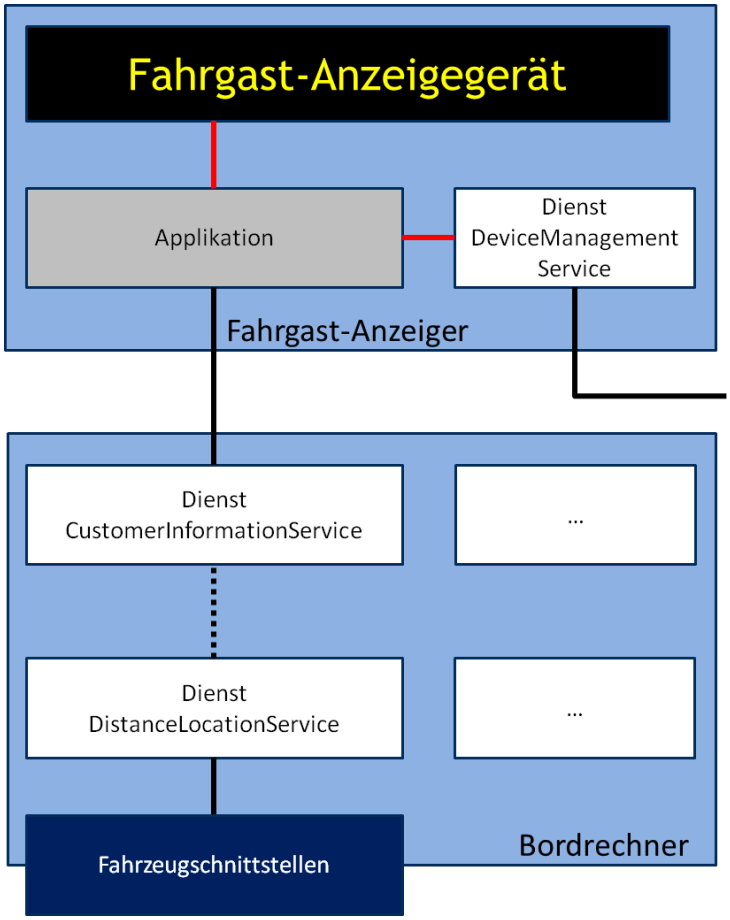


Abbildung 1 Beispiel zu den verschiedenen Repräsentationen von Fachkomponenten. Schwarze Linien: IBIS-IP, rote Linien: proprietär. Man beachte, dass die Fachkomponente Fahrgast-Anzeiger sowohl Geräte-, Applikations- wie auch Diensteigenschaften hat.

An dem System (vgl. Kapitel 1.3) 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.

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

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 3.3.1) 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.

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: `_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 2 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`

Best-Practice-Hinweis bezüglich der Host-Namen

Dienste, die auf demselben Gerät ausgeführt werden, melden den selben Hostnamen. Dies ermöglicht es dem Client, alle Dienste nach Hostnamen zu gruppieren.

3.3.1 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 1.5)
`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 5.1.4ff, vgl. auch Beispiel. in Kapitel 3.5)
`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 Zeit-synchronisation	`192.168.0.22`	Angabe der IP-Adresse unter welcher der SNTP-Server im IBIS-IP-Netzwerk zu finden ist.

Tabelle 3 Bedeutungen der TXT-Records in DNS-SD (inklusive Festlegungen für 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.

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>`

3 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), 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 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:

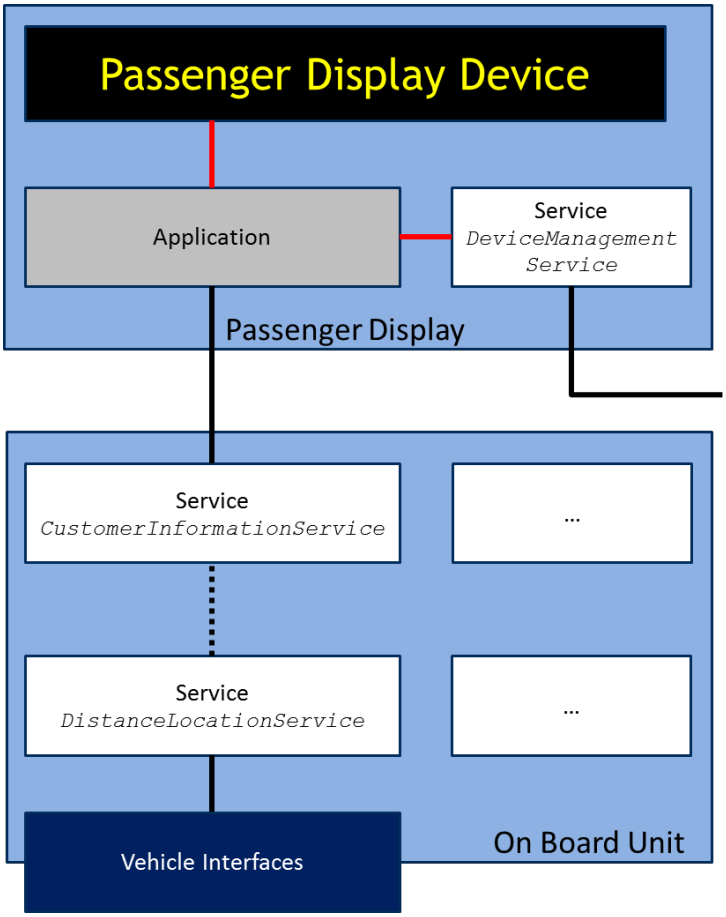


Figure 1: *Example for the different representations of a functional component. Black lines: IBIS-IP, red lines: proprietary. Please observe that the passenger display functional component has device properties, application properties, and service properties.*

Two devices participate in the system (cf. chapter 0):

— 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 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 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 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 2: 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`

Best Practice Note Regarding Host Names

Services running on the same device report the same hostname. This allows the client to group all services by hostname.

3.3.2 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 5.1.4pp, 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.
`coachnumber`	Mandatory in Trainset-IBIS-IP network	`IBIS-IP.string`	Number of the coach in which the service executing device is mounted. Distincts device information from different coaches
`deviceclass`	Mandatory from IBIS-IP 2.2 onwards	`see DeviceClassEnumeration`	IBIS-IP device class of the service executing device
`deviceID`	Mandatory from IBIS-IP 2.2 onwards	`IBIS-IP.NMTOKEN`	IBIS-IP device identification of the service executing device

Table 3: Meanings of the TXT record in DNS-SD (including the definitions for IBIS-IP)

3.4 Publication of UDP Services

UDP 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_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 Publication of HTTP Services

HTPP services indicate at least 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

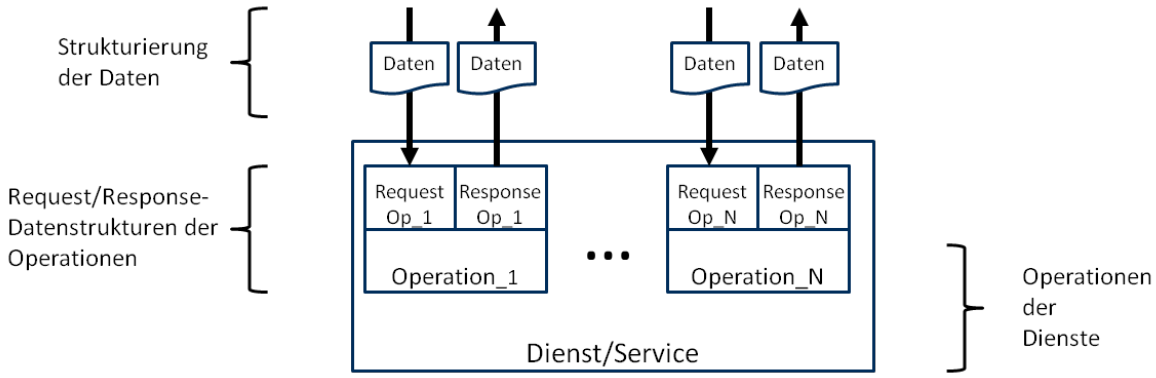


Abbildung 2 Darstellung der Strukturierung von Daten in Diensten

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

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.

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.

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.

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.

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 das Verb

— Restart.

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.

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.

4.1.2.1.1 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`

4.1.2.1.2 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`

4.1.2.1.3 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`

4.1.2.1.4 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`

4.1.2.1.5 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.

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.

Derzeit ist nur eine Operation, die das Restarten des Gerätes auslöst, vorgesehen.

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

4.1.2.2.1 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`***.

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`**

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.

Überwachung des Abonnements

Abonnements können überwacht werden. Spätestens nach Ablauf des Heartbeat-Intervall sendet der Service die zum Abonnement zugehörigen aktuellen Daten an den Abonnenten.

Das Heartbeat-intervall wird durch den Service festgelegt und bei der Bestätigung der Subcribe-Operation dem Abonnenten mitgeteilt (siehe **`SubscribeResponseStructure (Attribute Heartbeat))`**. Fehlt dieses Attribut in **`SubscribeResponseStructure`** (oder wird Heartbeat=0 ausgesendet), dann bietet der Server keine Abonnementüberwachung an.

Empfängt der Nachrichtenempfänger keine Daten innerhalb eines Heartbeat-Intervalls so kann er auf eine gestörte Verbindung schliessen und ggf. das Abonnement erneuern.

Das Heartbeat-Intervall wird mit einer Auflösung in Sekunden definiert, prinzipiell ist projektspezifisch ein angepasster Wert (z. B. im Minutenbereich) vorzusehen.

Spätestens mit Ablauf des Heartbeat-Intervalls sendet der Service die aktuellesten Daten an den Abonnenten. Prinzipiell ist es dem Service überlassen, ob er die Daten in einem festen Intervall versendet oder die Datenübermittelung ggf. optimiert.

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.

4.2 Konventionen für UDP-Dienste

Die UDP-Dienste veröffentlichen 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.

4.3 Konventionen für besonderes Dienstverhalten

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.
- **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.

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.

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>`

4 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.

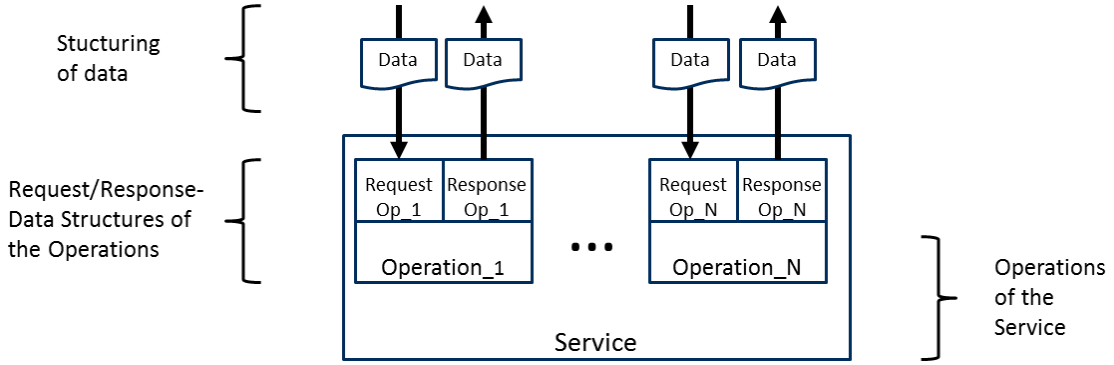


Figure 2: Representation of the structuring of data in services

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 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 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 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 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 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 verb is

— Restart.

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 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 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`**

4.1.2.1.2 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`

4.1.2.1.3 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`

4.1.2.1.4 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`

4.1.2.1.5 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 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.

Currently, only one operation is planned that triggers the restart of the device.

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

4.1.2.2.1 Trigger Restart

Operations used for restarting a process start with verb

Restart

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

4.1.2.3 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 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.

Subscription monitoring

Subscriptions can be monitored. At the latest when the heartbeat interval expires, the service sends out the current data associated with the subscription.

The heartbeat interval is determined by the service and communicated to the subscriber when the Subcribe operation is confirmed (see **SubscribeResponseStructure** (**Heartbeat** Attributes)). If this attribute is missing in **SubscribeResponseStructure** (or if **Heartbeat**=0 is sent), the server does not offer subscription monitoring.

If the message recipient does not receive any data within a heartbeat interval, it can conclude that there is a faulty connection and renew the subscription if necessary.

The heartbeat interval is defined with a resolution in seconds. In principle, a project-specific adapted value (e.g. in the minute range) must be provided.

At the latest when the heartbeat interval expires, the service sends the most current data to the subscriber. In principle, it is up to the service whether it sends the data at a fixed interval or whether it optimizes the data transmission.

4.1.4 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.

4.2 Conventions for UDP Services

The 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 Conventions for special Service Behavior

4.3.1 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.
— **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.

4.3.2 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 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 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.

5.1 Grundlagen

5.1.1 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 5.1.4*) 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.*

5.1.2 Fachliche Identifikation von Gerätem

Ein Gerät wird in IBIS-IP systemweit eindeutig durch

— einen Gerätetyp und
— eine Geräte-ID

identifiziert.

Mögliche Gerätetypen finden sich in Abschnitt 1.3 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.

5.1.3 Technische Identifikation von Geräten

Technisch kann ein Gerät innerhalb des Netzwerks über

— seine systemweit eindeutige IP-Adresse oder
— seinen systemweit eindeutigen DNS-Namen

identifiziert werden.

5.1.4 Fachliche Identifikation von Diensten

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. Allerdings ist es nicht IBIS-IP konform den gleichen Service in der gleichen Version auf einem Gerät mehrfach auszuführen.

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

5.1.4.1 Technische Identifikation von Diensten

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.*

5.2 Das Konzept des Systemstarts

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

1. Start der Dienste
   Jedes IBIS-IP Gerät startet, überwacht und beendet seine Dienste autonom.
2. Veröffentlichung der Dienste
3. Suche nach Diensten
4. Aufruf der Operationen der Dienste

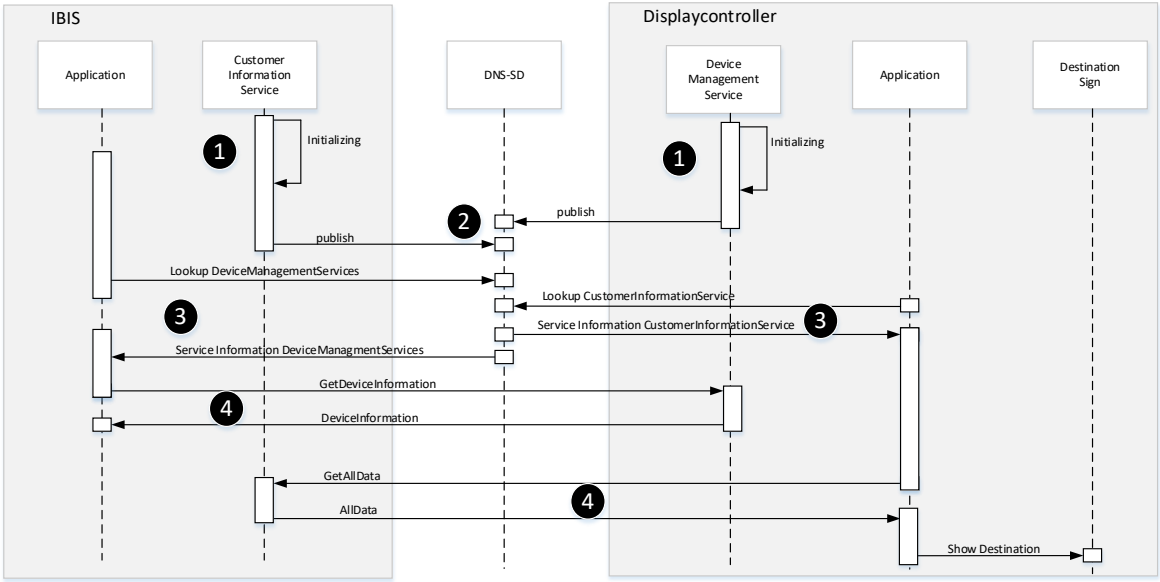
Beispiel:
1. Start der Dienste
Auf dem IBIS Bordrechner wird die Dienst `CustomerInformationService` *gestartet. Auf im Displaycontroller der* `DeviceManagementService`*.*

2. Veröffentlichung der Dienste
Die Dienste `CustomerInformationService` *und* `DeviceManagementService` veröffentlichen *die Dienstinformationen im DNS-SD*.

`3.` Suche nach Diensten
Die Applikation auf dem Bordrechner sucht über DNS-SD nach allen `DeviceManagmenService` *um GeräteInformation abzufragen und den Gerätestatus überwachen zu können. Die Applikation auf dem Displaycontroller sucht ihrerseits nach dem* `CustomumerInformationService` *um an die Anzeigedaten zu kommen.*

4. Aufruf der Operationen der Dienste
Die Applikation auf dem Bordrechner ruft die Operation GetDeviceInformation auf.
Die Applikation of dem DisplayController ruft die Methode GetAllData auf. Damit werden alle Daten des CustomerInformationService übertragen. Diese können nun aufbereitet und auf dem Anzeiger dargestellt werden.

Anmerkung: Im Beispiel sind nur die involvierten Komponenten dargestellt, beispielsweise ist im IBIS der DeviceManagementService nicht dargestellt.



5.2.1 Start der Dienste

Beim Start der Geräte eines IBIS-IP-Systems startet auf jedem der Geräte der Dienst `DeviceManagementService` (vgl. VDV 301-2-2) wie auch die fachlichen Dienste. Auf jedem Gerät des IBIS-IP-Systems läuft genau ein `DeviceManagementService`

5.2.2 Veröffentlichung der 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 diesem Dienst systemweit bekannt gemacht:

— Dienstname
— IBIS-IP-Version des Dienstes
— Protokoll, über das der Dienst angesprochen werden kann `_ibisip_http` oder `ibisip_udp`
— IP-Adresse bzw. DNS-Name des Gerätes, auf dem der Dienst läuft
— Port, unter dem der Dienst zu erreichen ist
— Pfad, unter dem der Dienst zu erreichen ist

5.2.3 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.

5 Principles in IBIS-IP

System start, the subsequent start of all services according to configuration, 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 Basics

5.1.1 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* 5.1.4*), 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.*

5.1.2 Functional Identification of devices

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 1.3.

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.

5.1.3 Technical Identification of devices

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 Functional Identification of services

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. However, it is not IBIS-IP compliant to run the same service in the same version several times on one device.

Example:

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

5.1.4.1 Technical Identification of Services

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 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:

1. Start services
   Each IBIS-IP device starts, monitors and terminates its services autonomously.
2. Publish services
3. search for services
4. call operations of services

Example:

1. start of the services

The CustomerInformationService service is started on the IBIS on-board computer. The DeviceManagementService is started on the display controller.

2. publication of the services

The CustomerInformationService and DeviceManagementService services publish the service information in the DNS-SD.

3. search for services

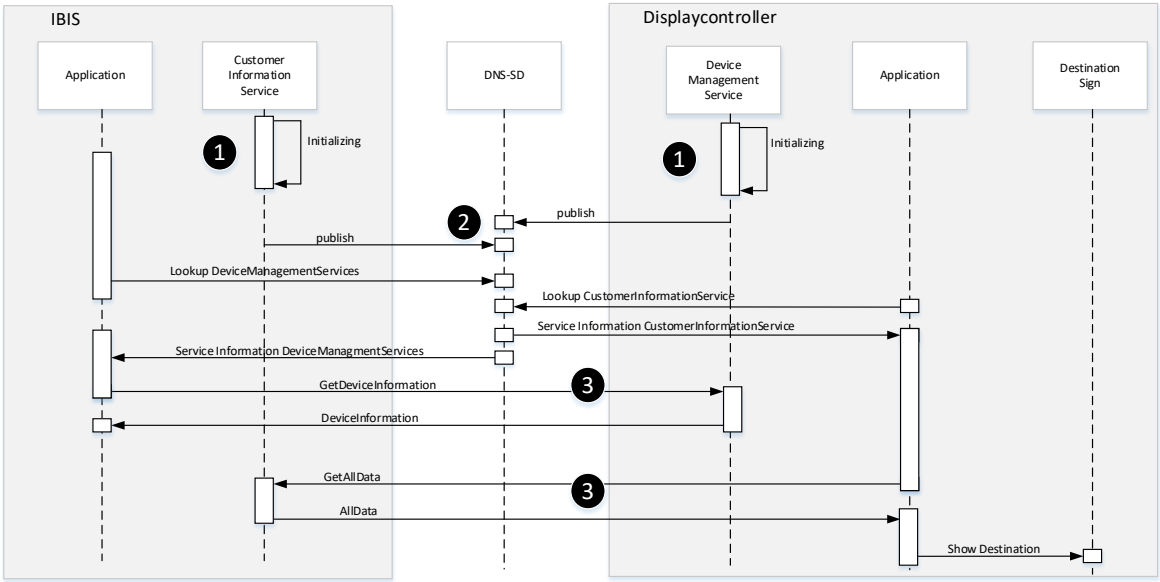
The application on the on-board computer searches for all DeviceManagmenService via DNS-SD in order to query device information and monitor the device status. The application on the display controller searches for the CustomumerInformationService to access the display data.

4. call the operations of the services

The application on the on-board computer calls the GetDeviceInformation operation.

The application of the DisplayController calls the GetAllData method. This transfers all data of the CustomerInformationService. These can now be processed and displayed on the display.

Note: In the example only the involved components are shown, for example the DeviceManagementService is not shown in IBIS.



5.2.1 Start of the Services

When the devices of an IBIS IP system are started, the `DeviceManagementService` (see VDV 301-2-2) as well as the specialist services start on each of the devices. Each device of the IBIS IP system runs exactly one DeviceManagementService.

5.2.2 Publication of the Services

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` or `ibisip_udp`
— 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

5.2.3 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.

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.

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 genutzen Tabellenform 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.

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.

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 4 Beispiel für Operationsnotation in IBIS-IP

6.1.3 Tabellennotation von XML-Strukturen

In dieser Schrift werden XML-Strukturen in einer Tabellennotation dargestellt (vgl. Tabelle 5). 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 5 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 5 Beispiel für die tabellarische Notation einer XML-Struktur

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.

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.

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“.

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.

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.

6 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 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 used table form.

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 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 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 4: Example of an operation notation in IBIS-IP

6.1.3 Table Notation of XML Structures

In this document, XML structures are represented in a table notation (cf. Table 5). 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 Table 5. 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 5: Example for the tabularized notation of a XML structure

6.1.3.1 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 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 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 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 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 Versionshistorie

7 Version History

7.1 Version 2.2

7.1.1 Funktionale Erweiterungen

- Neues Dokument basierend auf **301-2-sds-v2-1-basicservices erstellt**
- Folgende Änderungen gegenüber Version 2.1
    - Aufstartverhalten vereinfacht
    - Konkretisierung zu IBIS-IP Version
    - DeviceManagementService Operationen zur Aktivierung/Deaktivierung von Geräten und Diensten entfernt
    - Neue Geräteklasse ‚CombiDevice'
    - DNS-SD TXT-record erweitert
    - Best-Practice Hinweis zu Host-Namen hinzugefügt
    - SystemDocumentation Service entfernt
    - SystemManagement in SystemMonitoring umbenannt und in ein eigenes Dokument ausgelagert.

7.1.1 Functional Upgrade

- New document created based on 301-2-sds-v2-1-basicservices
- The following changes compared to version 2.1
    - Startup behavior simplified
    - Concretization to IBIS-IP Version
    - DeviceManagementService operations for activating/deactivating devices and services removed
    - New device class 'CombiDevice
    - DNS-SD TXT-record extended
    - Best practice note added to host names
    - SystemDocumentation Service removed
    - SystemManagement renamed SystemMonitoring and outsourced to a separate document

7.1.2 Technische Ergänzungen/Korrekturen

Keine

7.1.2 Technical Upgrade/Corrections

none

7.2 Version 2.3

7.1.3 Funktionale Erweiterungen

7.2.1 Functional Upgrade

7.1.4 Technische Ergänzungen/Korrekturen

In Kapitel 2.4 wurde die zu nutzende http Version ergänzt.

7.2.2 Technical Upgrade/Corrections

In chapter 2.4 the http version to be used was added.

8 Begriffe

Die bereits im VDV 301 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.
OnBordUnit	Entspricht der Geräteklasse des Zentralrechners aus IBIS Zeiten.
Other (Geräteklasse)	Hierunter lassen sich Geräte anbinden die noch nicht in diesem Standarad Klassifiziert wurden.

Abkürzung	Beschreibung
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.
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

8 Terms

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

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).

Term	Description
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.
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.
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 / Standards and recommendations

(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 (V1.0)
(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 / List of figures

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.	24
Abbildung 2	Darstellung der Strukturierung von Daten in Diensten	35
Figure 1:	*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.*	30
Figure 2:	Representation of the structuring of data in services	44

Tabellenverzeichnis / List of tables

Tabelle 1	IBIS-IP Geräteklassen Klassifikation (Sortierung beliebig)	10
Tabelle 2	Bedeutungen der SRV-Records in DNS-SD	26
Tabelle 3	Bedeutungen der TXT-Records in DNS-SD (inklusive Festlegungen für IBIS-IP)	27
Tabelle 4	Beispiel für Operationsnotation in IBIS-IP	63
Tabelle 5	Beispiel für Kapitel 1 für die tabellarische Notation einer XML-Struktur	64

Table 1:	IBIS-IP device class classification (any sorting)	14
Table 2:	Meanings of the SRV record in DNS-SD	32
Table 3:	Meanings of the TXT record in DNS-SD (including the definitions for IBIS-IP)	33
Table 4:	Example of an operation notation in IBIS-IP	67
Table 5:	Example for chapter1 for the tabularized notation of a XML structure	68

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