

International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

F.735.2

(06/2021)

SERIES F: NON-TELEPHONE TELECOMMUNICATION
SERVICES

Multimedia services

Architecture and protocols for software-defined cameras

Recommendation ITU-T F.735.2

ITU-T

ITU-T F-SERIES RECOMMENDATIONS
NON-TELEPHONE TELECOMMUNICATION SERVICES

TELEGRAPH SERVICE	
Operating methods for the international public telegram service	F.1–F.19
The gentex network	F.20–F.29
Message switching	F.30–F.39
The international telemesssage service	F.40–F.58
The international telex service	F.59–F.89
Statistics and publications on international telegraph services	F.90–F.99
Scheduled and leased communication services	F.100–F.104
Phototelegraph service	F.105–F.109
MOBILE SERVICE	
Mobile services and multideestination satellite services	F.110–F.159
TELEMATIC SERVICES	
Public facsimile service	F.160–F.199
Teletex service	F.200–F.299
Videotex service	F.300–F.349
General provisions for telematic services	F.350–F.399
MESSAGE HANDLING SERVICES	F.400–F.499
DIRECTORY SERVICES	F.500–F.549
DOCUMENT COMMUNICATION	
Document communication	F.550–F.579
Programming communication interfaces	F.580–F.599
DATA TRANSMISSION SERVICES	F.600–F.699
MULTIMEDIA SERVICES	F.700–F.799
ISDN SERVICES	F.800–F.849
UNIVERSAL PERSONAL TELECOMMUNICATION	F.850–F.899
ACCESSIBILITY AND HUMAN FACTORS	F.900–F.999

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T F.735.2

Architecture and protocols for software-defined cameras

Summary

As defined in Recommendation ITU-T F.735.1, software-defined cameras can provide the basic hardware service application programming interfaces (APIs) and the common software service APIs for application developers; these APIs are called "service-oriented interfaces". Through these APIs the upper layer applications can obtain raw video, pan/tilt/zoom (PTZ), common software and hardware resource to realize video intelligent analysis and non-intelligent businesses. Recommendation ITU-T F.735.2 specifies the architecture and service-oriented interface protocols for software-defined cameras, including functional architecture of software-defined camera system, service-oriented interface message protocol structure, and service-oriented interface protocols.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T F.735.2	2021-06-13	16	11.1002/1000/14678

Keywords

Service-oriented interface, software-defined camera.

* To access the Recommendation, type the URL <http://handle.itu.int/> in the address field of your web browser, followed by the Recommendation's unique ID. For example, <http://handle.itu.int/11.1002/1000/11830-en>.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents/software copyrights, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the appropriate ITU-T databases available via the ITU-T website at <http://www.itu.int/ITU-T/ipr/>.

© ITU 2021

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

Table of Contents

		Page
1	Scope	1
2	References.....	1
3	Definitions	1
	3.1 Terms defined elsewhere	1
	3.2 Terms defined in this Recommendation.....	1
4	Abbreviations and acronyms	2
5	Conventions	2
6	Architecture of an SDC system	2
	6.1 Functional architecture of an SDC system	3
	6.2 Service-oriented interface architecture and classification.....	4
7	SDC SOI message protocol structure	6
	7.1 SDC SOI access mechanism definition	6
	7.2 Message specification (HBTP).....	7
	7.3 SOI message description format.....	9
8	Protocols of service-oriented interface	9
	8.1 Video service API.....	9
	8.2 Algorithm inference API	20
	Appendix I – Data Type Definition and API Example.....	25
	I.1 SOI data type definition example	25
	I.2 Video service API example	26
	I.3 Algorithm inference example	34
	Bibliography.....	39

Recommendation ITU-T F.735.2

Architecture and protocols for software-defined cameras

1 Scope

Service-oriented interfaces are those APIs that are provided by the software-defined camera (SDC) operating system (OS) for the upper application developer to invoke the hardware and common software resources including raw video, algorithms inference, event, pan/tilt/zoom (PTZ), etc. to complete the intelligent analysis application development. The upper application layer can also obtain raw video, PTZ, common software and hardware resources through these APIs to complement intelligent analysis and non-intelligent business of monitoring. This Recommendation specifies the architecture and protocols for software-defined cameras, including the functional architecture of an SDC system, the service-oriented interface message protocol structure, and service-oriented interface protocols.

The scope of this recommendation includes:

- 1) Architecture of SDC systems;
- 2) SDC service-oriented interface (SOI) message protocol structure;
- 3) Protocols of service-oriented interface.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

- [ITU-T F.735.1] Recommendation ITU-T F.735.1 (2020), *Requirements for software-defined cameras*.
- [ITU-T F.743.1] Recommendation ITU-T F.743.1 (2015), *Requirements for intelligent visual surveillance*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following term defined elsewhere:

3.1.1 software-defined camera [ITU-T F.735.1]: Software-defined camera is a kind of IPU (see [ITU-T F.743.1]), which provides a technical approach to decouple hardware and software and to support algorithms' on-demand deployment, online upgrade without services interrupting, continuous self-adaptive learning to adapt to various scenarios.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 SDC studio: A software-defined camera (SDC) algorithm warehouse that is deployed in a cloud server and which supports service including centralized artificial intelligence model training,

intelligent algorithms online development, algorithm integration verification, application packing and release.

3.2.2 SDC controller: A client located in the front-edge of a video surveillance system, and which acts as an operation centre for massive software-defined cameras (SDCs). The main functions include SDC control (such as restart, stop, configuration, etc.) and algorithm management (such as deployment, upgrade, deletion, etc.).

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

API	Application Programming Interface
FSAAS	File System as a Service
HBTP	Hyper Binary Transfer Protocol
HTTPS	Hyper Text Transfer Protocol over Secure Socket Layer
NNIE	Neural Network Inference Engine
NoSQL	Nor only Structured Query Language
OS	Operating System
PTZ	Pan/Tilt/Zoom
SDC	Software-Defined Camera
SDK	Software Development Kit
SOI	Service-Oriented Interface
SQL	Structured Query Language
SSH	Secure Shell
VEnc	Video Encoder
VI	Video Input
YUV	luminance/luma (Y) – blue luminance (U) – red luminance (V)

5 Conventions

In this Recommendation:

- The keywords "is required to" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.
- The keywords "is recommended" indicate a requirement which is recommended but which is not absolutely required. Thus this requirement needs not be present to claim conformance.

6 Architecture of an SDC system

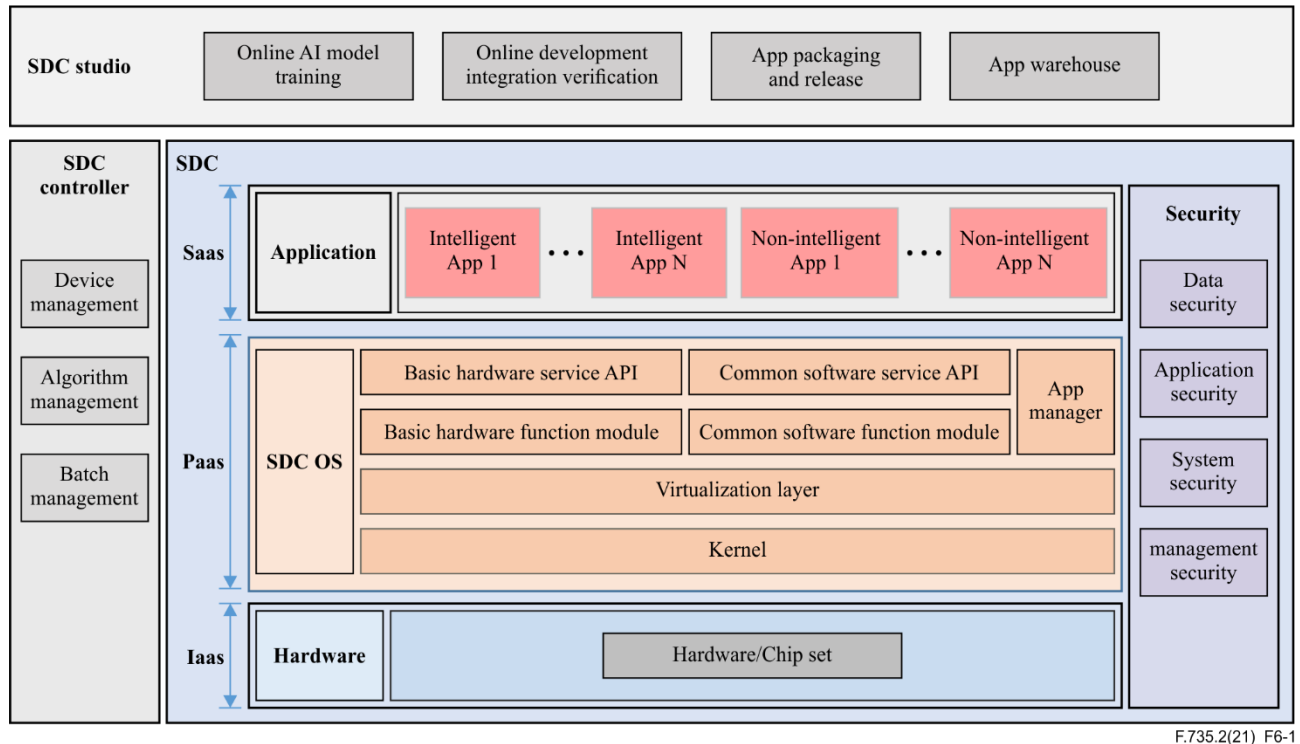
The overall functional architecture of SDC is defined in [ITU-T F.735.1]. This clause describes the whole functional architecture for an SDC system and the detailed service-oriented interface architecture.

6.1 Functional architecture of an SDC system

6.1.1 Overview of entities

The SDC system functional architecture could be decomposed into three units as follows. The architecture is illustrated in Figure 6-1.

- SDC studio (SDCS)
- SDC controller (SDCC)
- SDC



F.735.2(21)_F6-1

Figure 6-1 – Functional architecture of an SDC system

6.1.2 SDCCS: SDC studio

SDC studio can provide the algorithm developer a self-contained and efficient development platform, and can also provide the SDC user an algorithm warehouse from where the needed algorithm can be downloaded and deployed in specific SDC devices. The functions of such platform include AI model training, verification, online development, testing, etc. The detailed functions are as follows:

- AI model training and transfer: training AI model using the specific train data and providing AI model conversion tool to complete model conversion.
- Online algorithm development: supporting online algorithm development that is based on the SDC OS service APIs.
- Algorithm integration and verification: supporting specific algorithm tested and deployed in the test environment of SDC to verify the function of this algorithm.
- Algorithm packaging and release: providing tools to complete algorithm packaging.
- Algorithm warehouse: supporting app licence management, app reviewing and app billing, etc.

6.1.3 SDCC: SDC controller

The SDC controller is a client deployed in the front-end of the surveillance system. It creates connections with SDCs and SDC studio, which acts as an operation centre for SDC management,

including camera configuration, user management, device control, and algorithm life cycle management. It is recommended that the whole management operation support batch operation. The detailed functions are as follows:

- Algorithm download: accessing the SDC studio and downloading a specific algorithm.
- Device control: video live streaming, PTZ control, image capture, recording, etc.
- User management: identification and authentication, access control, authorization, etc.
- Device configuration: protocol access parameter configuration, alarm configuration, PTZ configuration, basic video streaming configuration, etc.
- Algorithm life cycle management: algorithm deployment, activating, deactivating, upgrade and deletion in SDC.

6.1.4 SDC

The architecture of SDC is already defined in [ITU-T F.735.1], and the security function is added in Figure 6-1 to provide a full explanation of SDC. The security management contains data security, application security, system security and management security.

The security management components provide:

- Data security: data integrity verification, encrypted storage of sensitive data, system data backup, key security, etc. The data type including image, intelligent metadata (such as vehicle attribute, object detection result, the extraction feature of a human body, etc.) and alarm data in the SDC.
- Application security: installation package (APP) integrity verification, application authorization and authentication, application licence management, etc.
- System security: provision of the necessary measures to achieve system security. This includes performing authority management and control on SDC key data to prevent users from unauthorized operation, providing a digital signature and verification tool for algorithm package integrity verification and anti-hacking.
- Management security: supporting remote maintenance security (such as using hypertext transfer protocol over secure socket layer (HTTPS) and secure shell (SSH) protocol when logging into in the SDC, during sensitive data transmission, etc.) and ensuring account and password security.

For the functions of SDC hardware, SDC OS, and application layer, please refer to clause 7 in [ITU-T F.735.1].

6.2 Service-oriented interface architecture and classification

6.2.1 Service-oriented interface architecture

The software-defined camera (SDC) software architecture consists of the following layers, which are illustrated in Figure 6-2:

- Customized service software layer: Provides a variety of service functions required by end users and supports on-demand service loading.
- Common software capability service layer: Provides support for the service software layer to accelerate the production of various customization scenarios and innovative services. The common software capabilities can be loaded on demand.
- Basic hardware capability service layer: Provides interfaces for using underlying hardware resources. All basic hardware capabilities at this layer are based on the SDC hardware capability.

- SDC service-oriented interface layer: Defines the specifications of communication interfaces among capability service layers to achieve robust simplicity, high efficiency and excellent scalability.

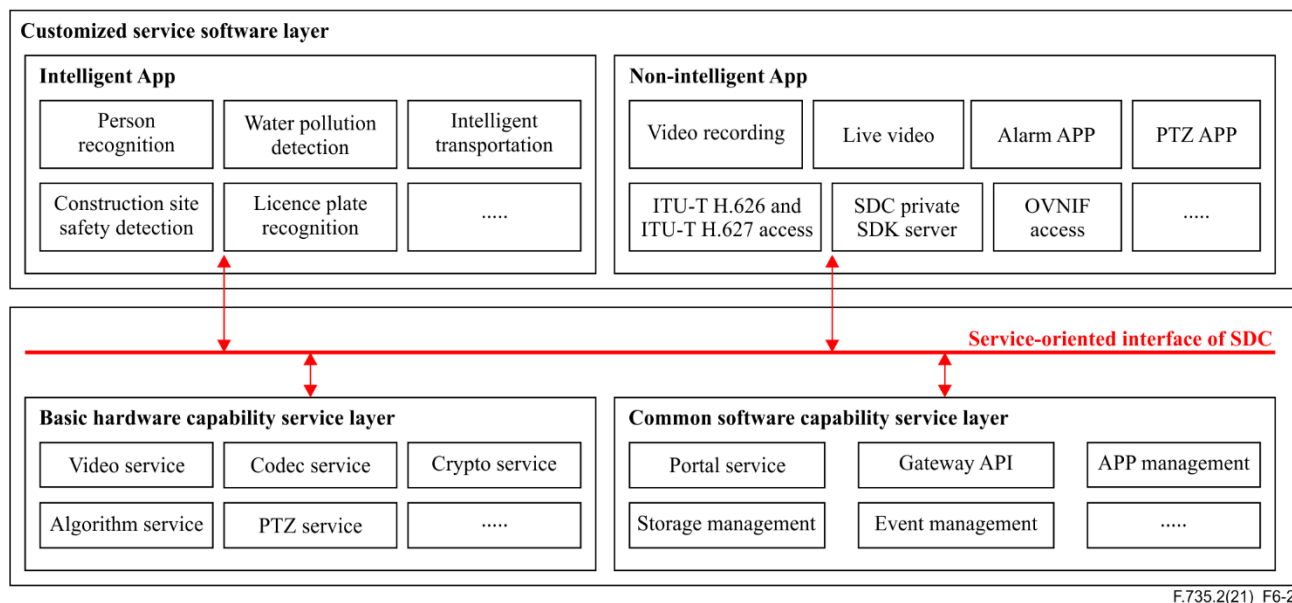


Figure 6-2 – Architecture of SDC software

6.2.2 Service-oriented interface classification

The SDC service-oriented interface can be classified into two types: the basic hardware capability service APIs and the common software capability service APIs.

Basic hardware capability service APIs

The basic hardware capability service APIs provide interfaces for using SDC hardware resources, including but not limited to getting the raw video (such as YUV frame), invoking the algorithm model inference, controlling the SDC PTZ, etc. For the function description of these service, see Table 6-1.

NOTE – YUV is abbreviation for "Luminance (Luma) / blue luminance (U) / red luminance (V)" that is a type of colour encoding representation historically used in analogue systems, such as colour components in SECAM and PAL colour spaces.

Table 6-1 – Basic hardware capability service interfaces

Service name	Function description
Video service	Provides the function of subscribing to intelligent algorithm input data – video stream data
Codec service	Provides the codec function for media chips
Crypto service	Provides the encryption and decryption functions supported by the bottom-layer chip
Algorithm service	Provides the function of loading and calculating algorithm model files for AI chips
PTZ control service	Provides the pan-tilt-zoom (PTZ) control function

Common software capability service APIs

The common software capability service APIs are used to provide the common software capabilities of the camera which include API gateway interface, event management, security, log, etc. For the function description of these services, see Table 6-2.

Table 6-2 – Common software capability service interfaces

Service name	Function description
App management	Provides app lifecycle management capabilities, such as app installation, uninstallation, enabling, disabling, starting, stopping, and watchdog.
Event management	Provides flexible event subscription mechanisms, and implements common capabilities of releasing and subscribing to event data based on the zero-copy mechanism.
Storage management	Provides the functions of managing disk partitions and storing NoSQL and SQL statements.
Portal service	Provides a unified entry for web pages of all SDC apps and the SDC OS. The entry is a human-machine interface.
Gateway API	Provides a unified entry for northbound software development kits (SDKs) of all SDC apps and the SDC OS. The northbound SDK provides the service from SDC to client.

7 SDC SOI message protocol structure

7.1 SDC SOI access mechanism definition

This clause specifies the invocation mechanism of service-oriented interfaces (SOIs). This invocation are instructions for using a file operating interface that is called file system as a service (FSAAS). All the methods are illustrated in Table 7-1. A user can invoke the **open** interface to open the file corresponding to a service and obtain the read/write handle. Then user can invoke the **read/write** interface to interact with the service. The operation for invoking the **open** interface is similar to that for setting up a TCP socket link. All the service files are located in the "/mnt/srvfs" directory in the camera system, and the service name is the file name. For example, the raw video service provided by the basic hardware capability service layer is located in the "/mnt/srvfs/video.iaas.sdc" file, this file directory address is the service URI of this raw video service interface.

Table 7-1 – The invocation methods of SOI

Methods	Description
Open	Get service handle
Read	Get service response
Write	Execute operation of interface
Close	Close service handle

The example of invoking the video service API which contains the raw video (YUV frame data) registration and subscription is shown in Figure 7-1, the variable "serviceHandle" is the video service handle, while the methods "open, write, read and close" are used to operate the service handle like file system operation. This flow is the invocation mechanism of SDC service-oriented interface.

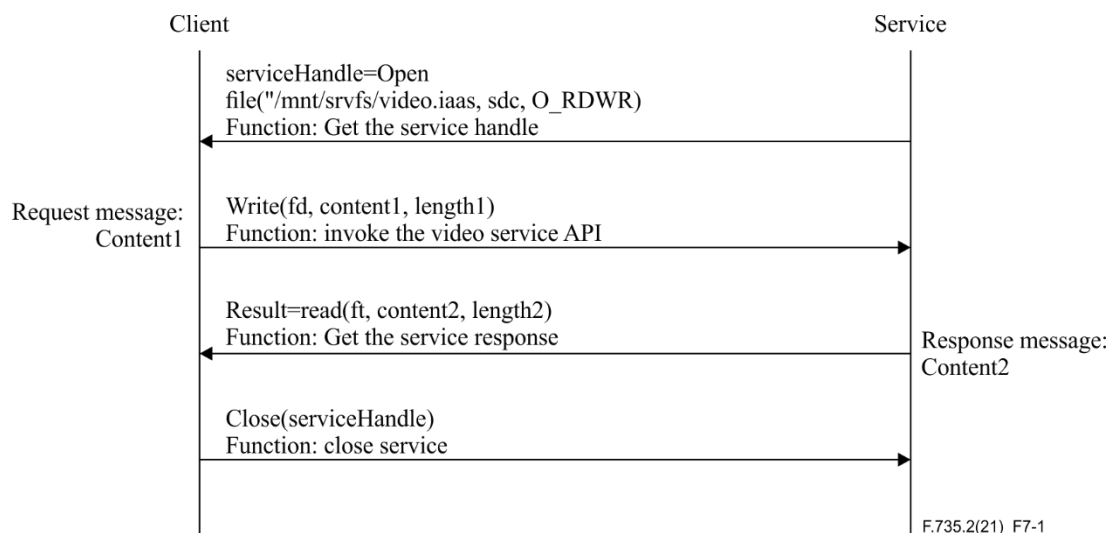


Figure 7-1 – Procedure flow of raw video service invocation

7.2 Message specification (HBTP)

This clause defines the message specification named hyper binary transfer protocol (HBTP) of such SDC SOI, which is shown in Figure 7-2. A message contains a maximum of three parts: common header, extension header and message content. The extension header and message content are optional. For details about the message content and extension header of each service interface, see the corresponding interface definition subclause in clause 8.

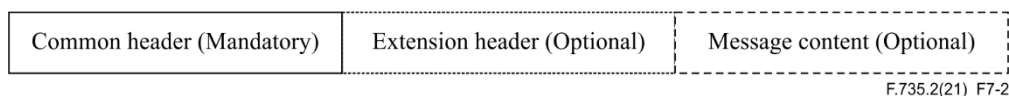


Figure 7-2 – HBTP of SDC SOI

NOTE – The message structures of the following service-oriented interfaces are defined according to the network order (high bytes stored in lower memory locations).

Common header

The common header is defined according to the network order (big endian) as shown in Figure 7-3.

version (2 bytes)	url-ver (1 byte)	R	method (7 bits)
url (2 bytes)	code (2 bytes)		
head_length (2 bytes)	trans-id (2 bytes)		
content_length (4 bytes)			

F.735.2(21)_F7-3

Figure 7-3 – Common header definition of SDC SOI

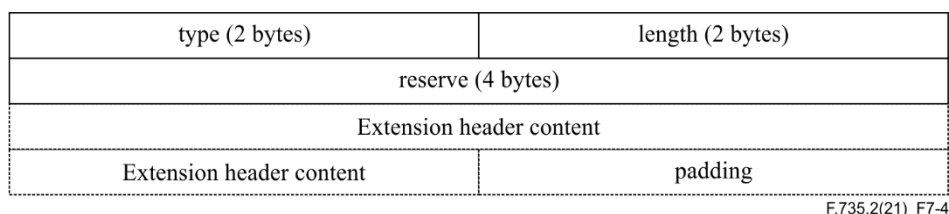
The definition of common header fields is shown in Table 7-2. The data structure can be defined based on the endian macro definition of the GCC; for an example, please refer to clause I.1.1.1.

Table 7-2 – Common header field definition

Field	Length	Semantics
version	2 bytes	Protocol version. Currently, the field has the fixed value of 0x5331 .
url_ver	1 byte	Interface version compatibility.
R	1 bit	Indicates whether a message is a request or response. The value 1 indicates response.
method	7 bits	Method definition. 1: CREATE 2: GET 3: UPDATE 4: DELETE
url	2 bytes	Resource ID
code	2 bytes	Response code (valid only when R is set to 1)
head_length	2 bytes	Total length of the common header and extension header
trans-id	2 bytes	Transaction ID. The trans_id field in the service response header corresponds to the trans_id field in the request. In the pipeline scenario similar to HTTP, the server cannot ensure that the sequence of the response and request is consistent. The client can match the request and the response based on this field.
content_length	4 bytes	Total length of the message content

Extension header

The specific type of the extension header and the definition of the data structure corresponding to this type of extension header are determined by each service interface defined in clause 8. The total length is aligned by 8 bytes. The extension header definition is illustrated in Figure 7-4.

**Figure 7-4 – Extension header definition of SDC SOI**

The definition of extension header fields is shown in Table 7-3. The data structure can be defined based on the endian macro definition of the GCC, for the example please refer to clause I.1.2.

Table 7-3 – Extension header field definition

Field	Length	Semantics
type	2	The specific type is defined by each service
length	2	Total length of the extension header and its content, excluding the padding character.
Extension header content	hdr_len - 8	Actual message body content carried
padding	$((\text{hdr_len} + 7) \& \sim 7) - \text{hdr_len}$	Ensure that the data of the entire extension header is 8-byte aligned

7.3 SOI message description format

The SOI message description is illustrated in Table 7-4. The field URI is the "url-ver" field in common header, the detailed definition of common header and extension can refer to clause 7.2. The methods CREATE, GET, UPDATE and DELETE are used in common header to complete the corresponding operation to SOI services. The message content is the specific message as defined in clause 8.

Table 7-4 – SOI message description format

Function	The interface function description				
URI	URI Description			URI Value	
Request				Response	
Common header		Extension header	Content	Extension header	Content
Method					
Response code					

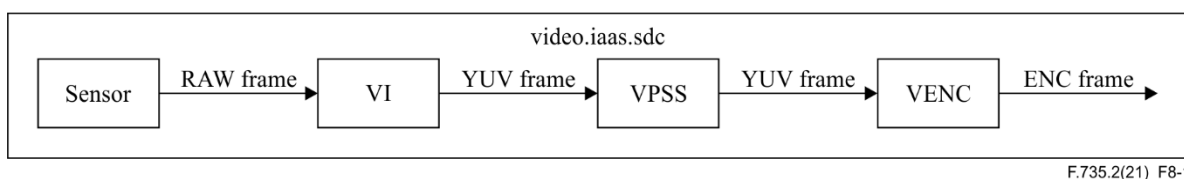
8 Protocols of service-oriented interface

This clause defines the video service APIs and the algorithm APIs which are essential APIs for algorithm developer to get raw video frame data and execute algorithm inference. With the algorithm inference APIs, the developer can invoke the deep learning inference ability of the AI chip and realize neural network model inference.

8.1 Video service API

8.1.1 Video service function definition

The processing of the raw video which is generated from a sensor is illustrated in Figure 8-1. The raw video is changed to a YUV frame after the processing in the video process function model, and after encoding the video data is converted to ENC frame data. The video service is aimed to support the YUV or video encoder (VENC) video data subscription capability. This clause specifies the YUV and VENC video data subscription, release function and image capture function.



F.735.2(21)_F8-1

Figure 8-1 – Raw video processing procedure in SDC

8.1.2 Video logical channel definition

There are two types of channels. One is the YUV video frame data channel that is not compressed. The other is the VENC frame (H.264/H.265) channel. Users can query the attributes and usage status of each channel and set the data output format depending on their needs. If there are multiple users, the users need to collaborate with each other to eliminate potential channel resource and configuration conflicts.

8.1.2.1 YUV logical channel definition

For the definition and channel number value of YUV logical channel, see Table 8-1.