SMPTE RP 2021-6:2017

SMPTE RECOMMENDED PRACTICE

BXF SDK Documentation

Table of Contents

Approved
July 12, 2017

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices, and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in its Standards Operations Manual. This SMPTE Engineering Document was prepared by Technology Committee 34CS

Intellectual Property

At the time of publication no notice had been received by SMPTE claiming patent rights essential to the implementation of this Engineering Document. However, attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. SMPTE shall not be held responsible for identifying any or all such patent rights.

Introduction

This section is entirely informative and does not form an integral part of this Engineering Document.

When the BXF suite of documents (SMPTE 2021-x) was first published, interoperability issues were sometimes encountered between systems because of different communication schemes, protocols and security constraints.

This Recommended Practice aims to solve these problems, specifically between Traffic and Automation systems. Automatic serialization classes are utilized for working with the BXF xml-based messages, which provides a baseline solution for the system interoperability.

The BXF SDK supports 3 types of transmitted objects between Traffic System and Automation:

1) Native BXF xml-based messages
2) Deserialized generic objects
3) Automation-friendly objects – an event and a list of events

An implementer can select any type to exchange data between the Traffic system and Automation system.

What follows is a basic model of the BXF SDK, which includes three layers between Traffic System and Automation System:

1. Traffic SDK Endpoint
2. Messages Bus (Mule ESB). The Mule ESB is utilized as the transport bus because of its simplicity and flexibility.
3. Automation SDK Endpoint

See figs below.

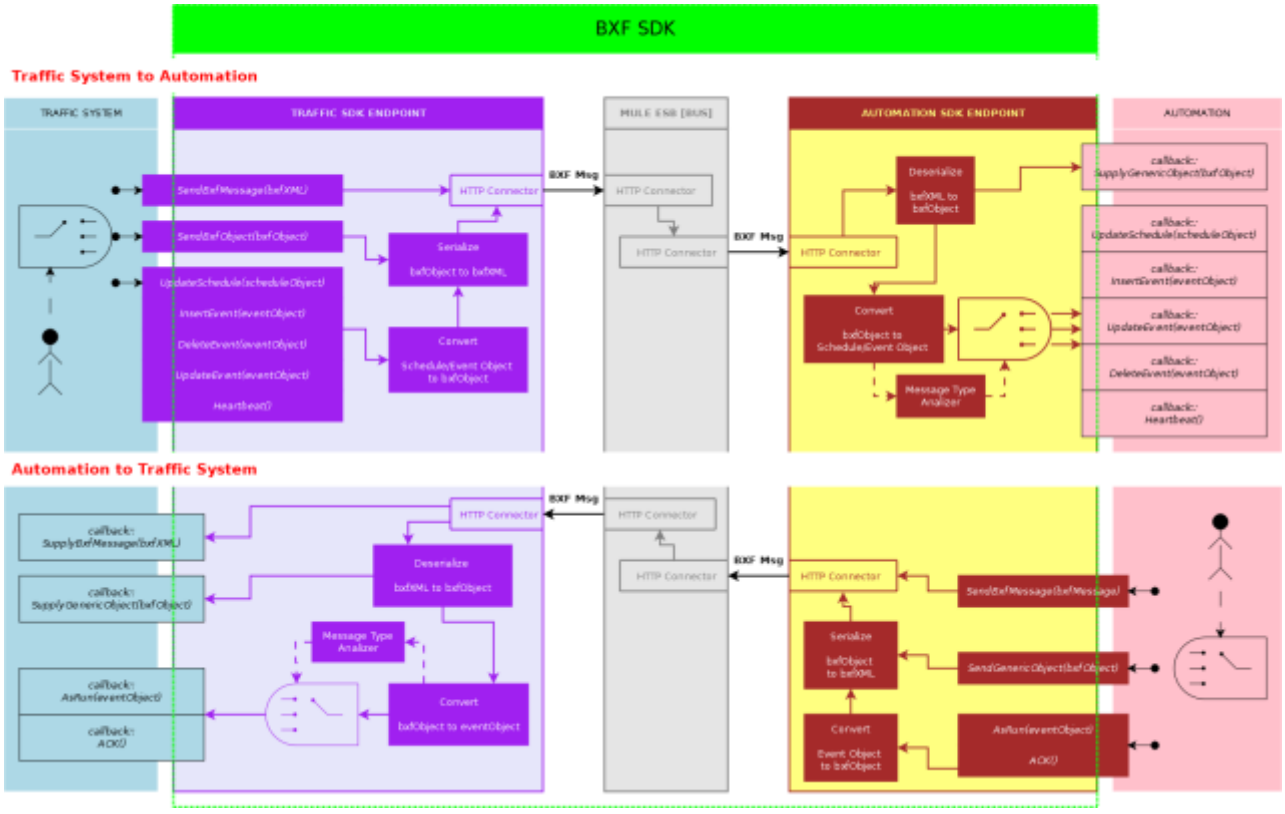


Fig 1. Basic Model

Direction	Traffic System → Automation	Automation → Traffic System
Module		
TRAFFIC SDK ENDPOINT	A collection of procedures to report a traffic schedule list changes	Callback procedures to send messages from Automation

BUS (Mule ESB)	Redirect xml messages from traffic endpoint to automation endpoint	Redirect xml messages from automation endpoint to traffic endpoint
AUTOMATION SDK ENDPOINT	Callback procedures from Automation to supply data from traffic system	A collection of procedures to report a automation play list changes

Fig 2. High-level API solutions

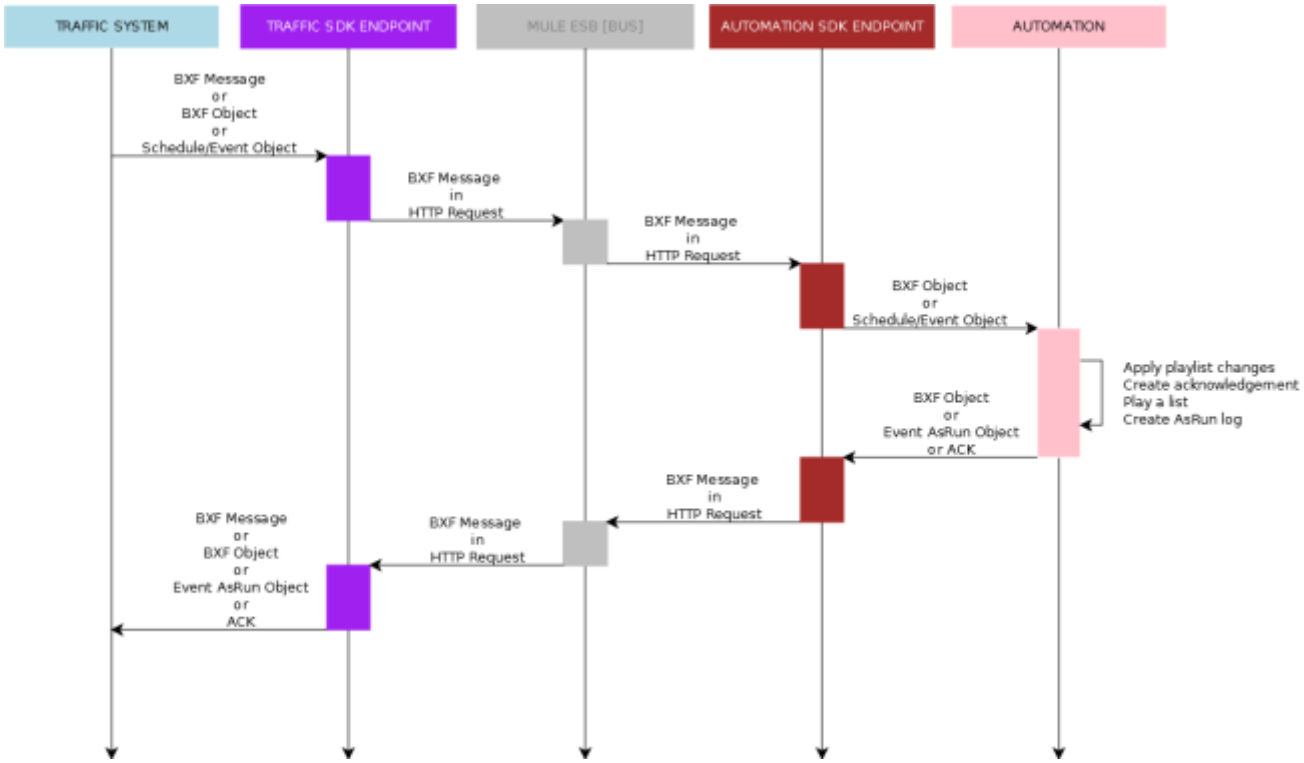


Fig. 3 Basic interaction workflow

1 Scope

This Recommended Practice provides documentation of the BXF SDK, a set of DLLs provided alongside this document, intended to facilitate improved interoperability among Traffic and Automation Systems.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful

to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword “reserved” indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword “forbidden” indicates “reserved” and in addition indicates that the provision will never be defined in the future.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

Unless otherwise specified, the order of precedence of the types of normative information in this document shall be as follows: Normative prose shall be the authoritative definition; Tables shall be next; then formal languages; then figures; and then any other language forms.

3 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this engineering document. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this engineering document are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

There are no normative references

4 Terms and Definitions

For the purposes of this document, the following terms and definitions apply.

4.1 Traffic System (TS)

A system used in the construction of playout schedules.

4.2 Automation System (AS)

A system used in the on-air execution of playout schedules.

4.3 Traffic Agent (TA)

An SDK module processing on TS side.

4.4 Automation Agent (AA)

An SDK module processing on AS side.

4.5 Associate System

TS for TA and is AS for AA

4.6 Agent

Can refer to a Traffic Agent or Automation Agent.

4.7 Partner

Contra-side system or agent : Traffic Agent or Traffic System for a corresponding Automation Agent, or an Automation Agent or Automation System for a corresponding Traffic Agent.

4.8 Validator

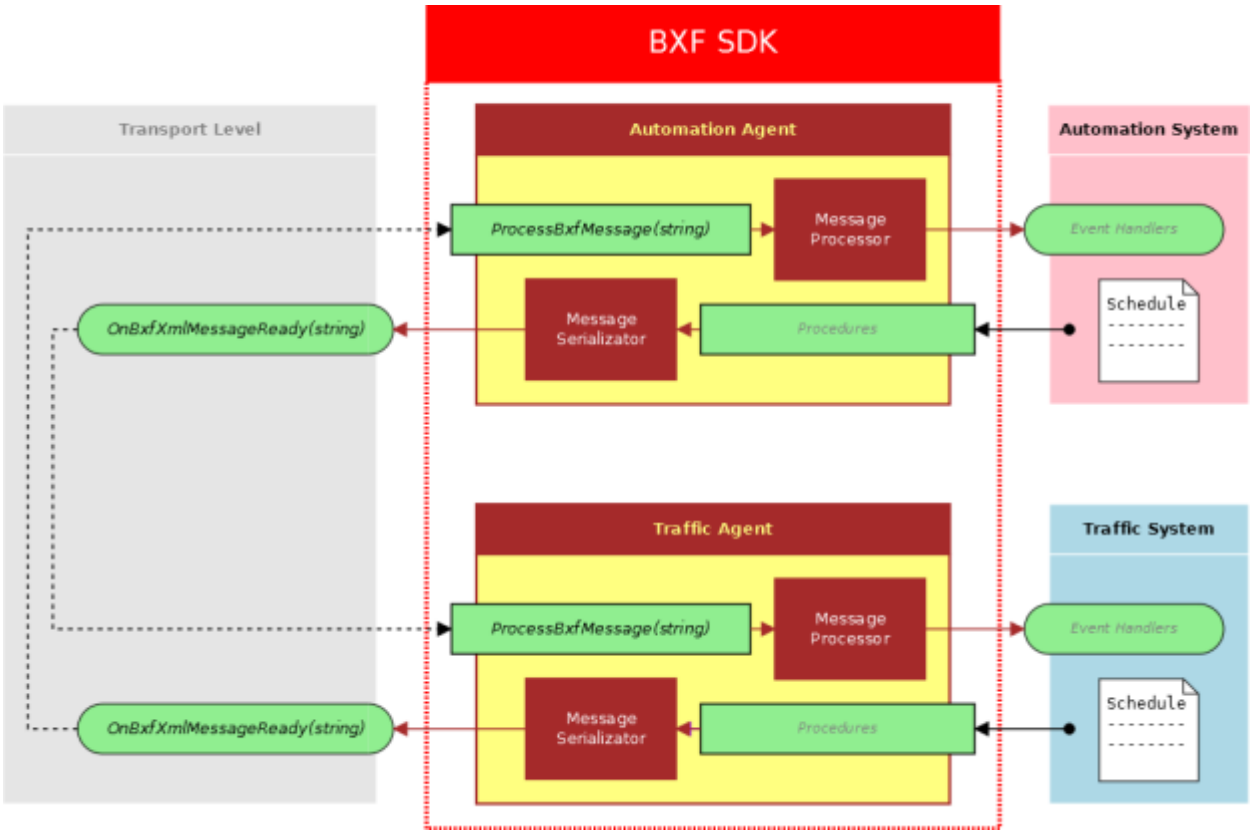
An SDK module used for validating BXF XML messages against xsd schema

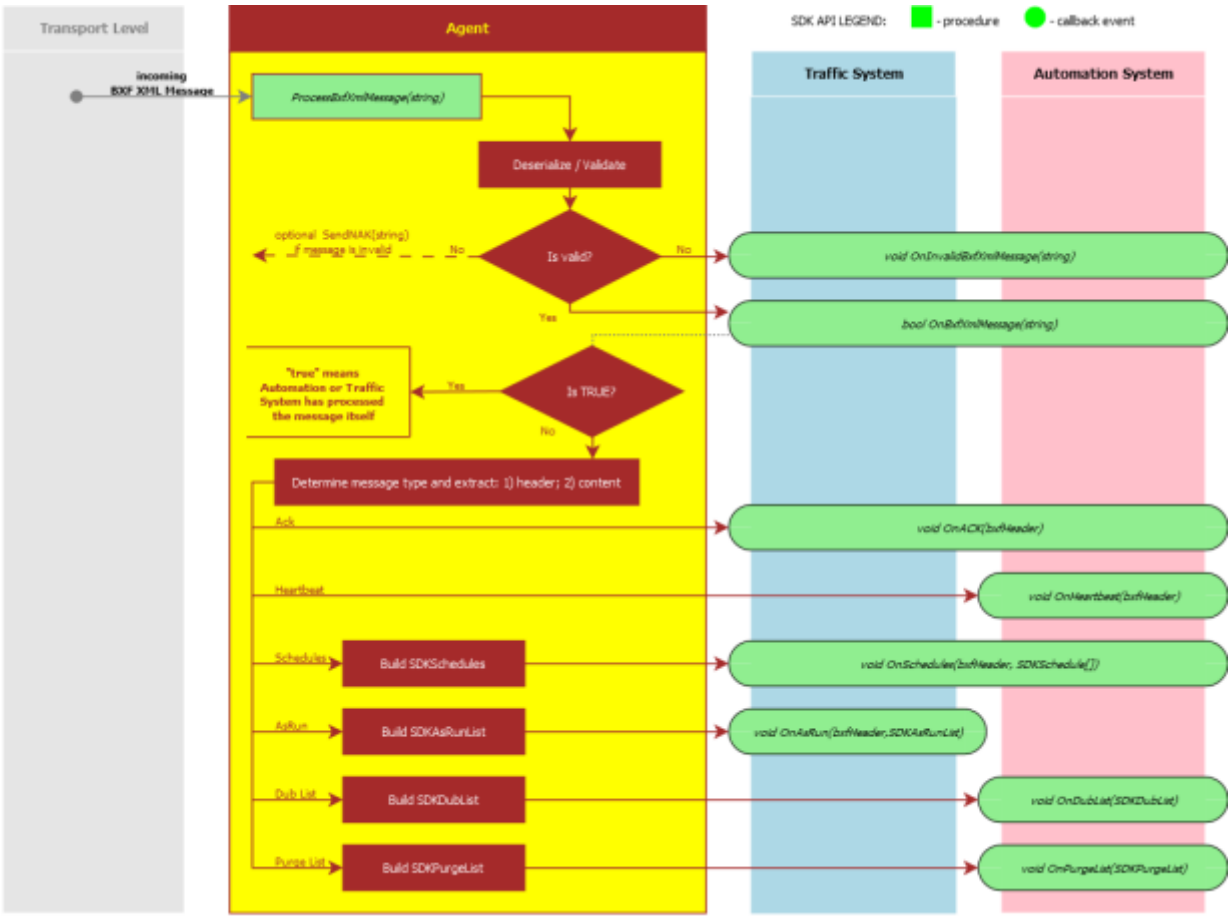
4.9 SDKBox

An SDK high level module used to hold and manage SDK modules (Transport Adapters, Agent, Validator)

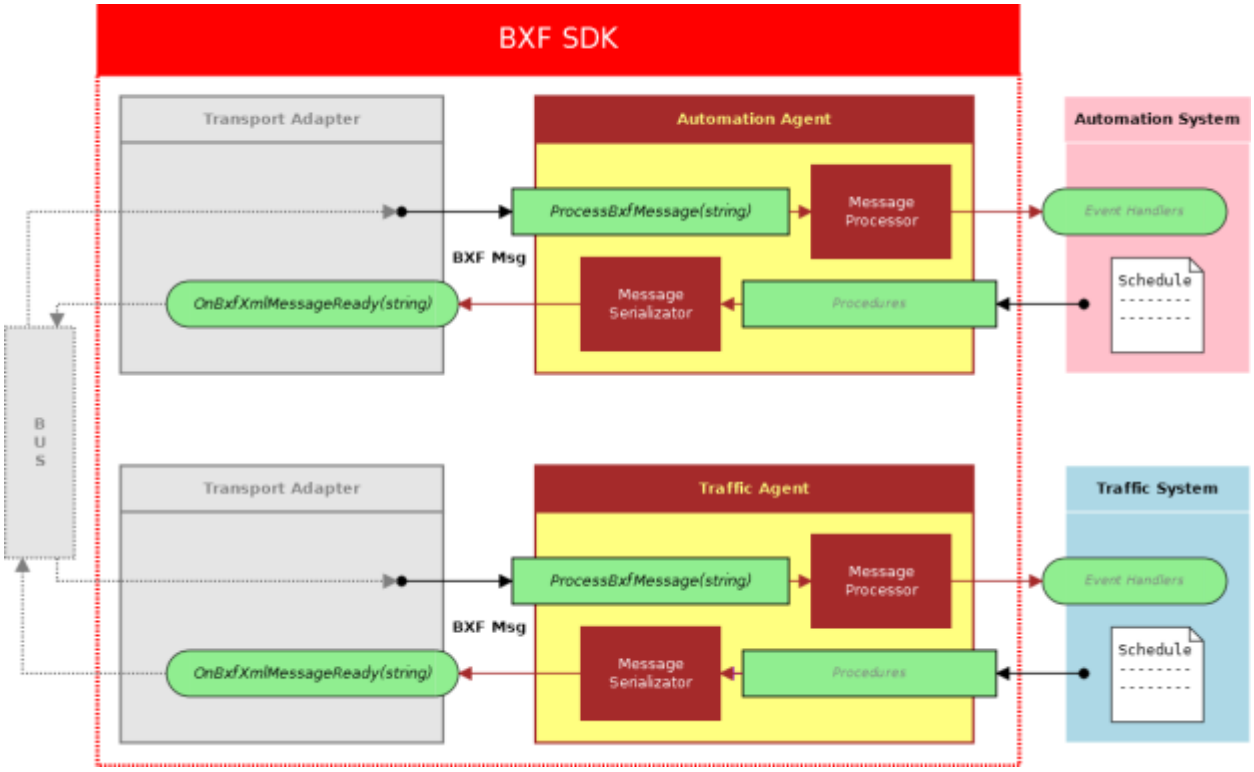
5 The SDK Structure

5.1 With external transport level




Transport Level
Agent
SDK API LEGEND:
- procedure
- callback event
Traffic System
Automation System
incoming
BXF XML Message
ProcessBxfXmlMessage(string)
Deserialize / Validate
optional SendNAK(string)
_if message is invalid
No
Is valid?
No
void OnInvalidBxfXmlMessage(string)
Yes
bool OnBxfXmlMessage(string)
"true" means
Automation or Traffic
System has processed
the message itself
Yes
Is TRUE?
No
Determine message type and extract: 1) header; 2) content
Ack
void OnACK(bxfHeader)
Heartbeat
void OnHeartbeat(bxfHeader)
Schedules
Build SDKSchedules
void OnSchedules(bxfHeader, SDKSchedule[])
AsRun
Build SDKAsRunList
void OnAsRun(bxfHeader,SDKAsRunList)
Dub List
Build SDKDubList
void OnDubList(SDKDubList)
Purge List
Build SDKPurgeList
void OnPurgeList(SDKPurgeList)

5.2 With embedded transport adaptor

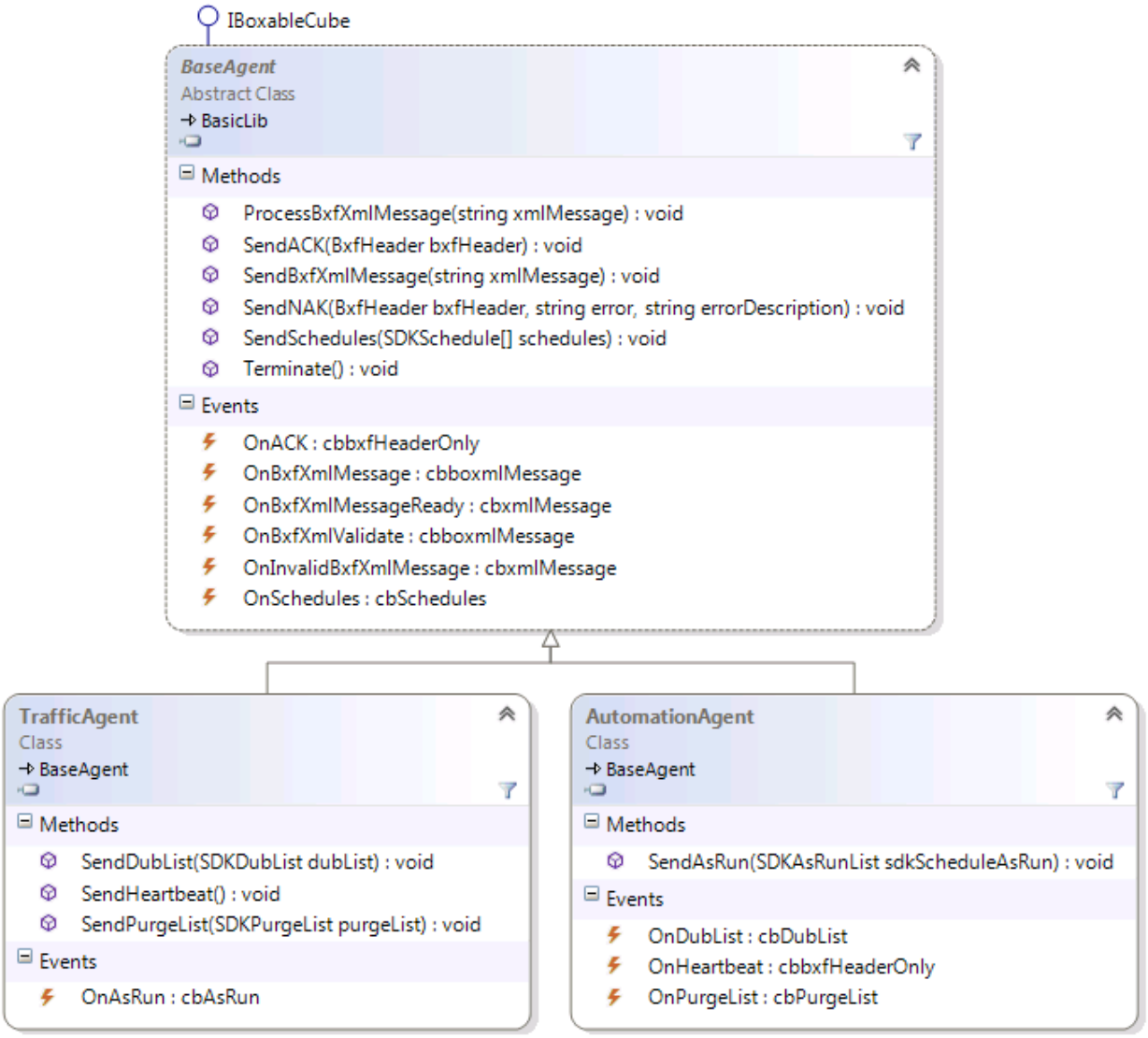


5.3 Agents

Traffic/Automation Agent – BXF SDK module which interacts with Traffic/Automation System.

The Agents:

- receive bxf messages from transport level (Transport Adapter or another implementation)
- validate and deserialize bxf message
- call event handlers from Traffic System and provide received data as
    a) original raw xml string
    b) deserialized BXF object
    c) simplified SDK object for events and schedules (SDKEvent/SDKSchedule types)



Method:	Terminate()	Safe terminating

To Partner:

Event:	OnBxfXmlMessageReady	Occurs when an agent has got bxf message and is ready to give it to a Partner
Method:	ProcessBxfXmlMessage(string xmlMessage)	Receives and processes the message from a Partner.

To Associate System:

Events:	OnBxfXmlMessage	Occurs when an agent has got raw BXF message and is ready to give

		one to its Associate System. Event handler (callback function) should return boolean value: *true* – if Associated System processes BXF message itself, *false* – if Associated System does not process BXF message.
	OnBxfMessageValidate	Occures when an agent needs to validate BXF XML message. Validator event handler (callback function) should return boolean value: *true* – if BXF message is valid, *false* – otherwise.
	OnInvalidBxfMessage	Occures when an agent determines that received BXF message is not correct.
	OnACK	Occurs when an agent receives ACK BXF message from a partner
	OnSchedule	Occures when an agent receives a schedule from a partner
Methods:	SendBxfXmlMessage(string xmlMessage)	Sends raw BXF XML string *xmlMessage* from Associate System. Return nothing.
	SendSchedules(SDKSchedule[] schedules)	Sends array of schedules in SDKSchedule format from Associate System Returns nothing.
	SendACK(BxfHeader bxfHeader)	Sends Acknowledgement from Associated System for received message with *bxfHeader* header. Returns nothing.
	SendNAK(BxfHeader bxfHeader, string error, string errorDescription)	Sends Error Acknowledgement from Associated System for received message with *bxfHeader* header. *error* – short error definition *errorDescription* – detailed error description Returns nothing.

Traffic Agent only

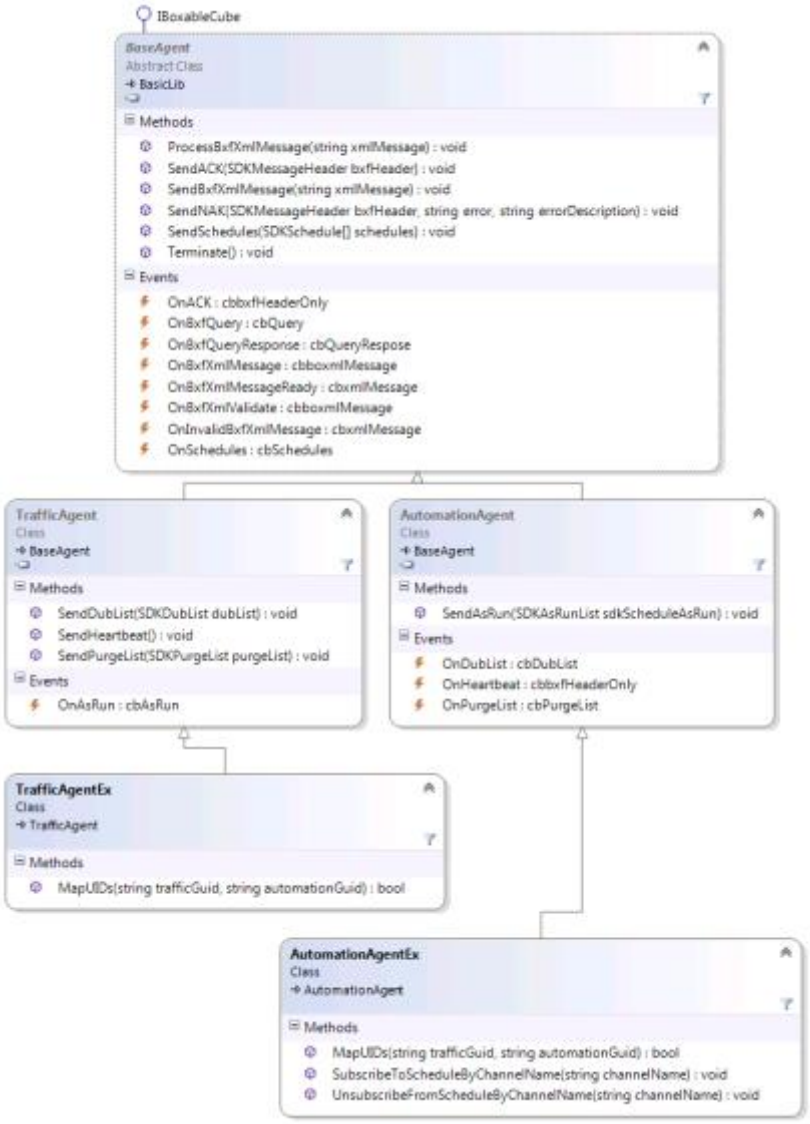
Event:	OnAsRun	Occurs when a TA receives AsRun BXF message.
Method:	SendHeartbeat	Creates and sends heartbeat BXF message to Partner.
	SendDubList(SDKDubList dubList)	Sends Duplication content transfer message
	SendPurgeList(SDKPurgeList purgeList)	Sends Purge content transfer message

Automation Agent only

Event:	OnHeartbeat	Occurs when AA receives heartbeat BXF message.
	OnDubList	Occures when an automation receives a duplication content transfer list from traffic system
	OnPurgeList	Occures when an automation receives a purge content transfer list from from traffic system

Method:	SendAsRun(SDKAsRunList asRunList, bool complete)	Sends AsRun list in SDKAsRunList format to Partner, If *complete* is *false*, AsRunList items are in short basic form, *otherwise* in extended complete form Return nothing.

5.4 Extended Agent



Extra features which are implemented in extended version:

1. Event GUID substitution

2. Subsciption to selected channel

Extended Traffic Agent

Method:	**MapUID(string trafficGuid, string automationGuid)**	**Map event traffic ID to automation ID**

Extended Automation Agent

Method:	**MapUID(string trafficGuid, string automationGuid)**	**Map event automation ID to traffic ID**
	SubscribeToScheduleByChannelName (string channelName)	Subscribe an agent to process schedules for seleted channels only
	UnsubscribeFromScheduleByChannelName (string channelName)	Remove subscription to channel

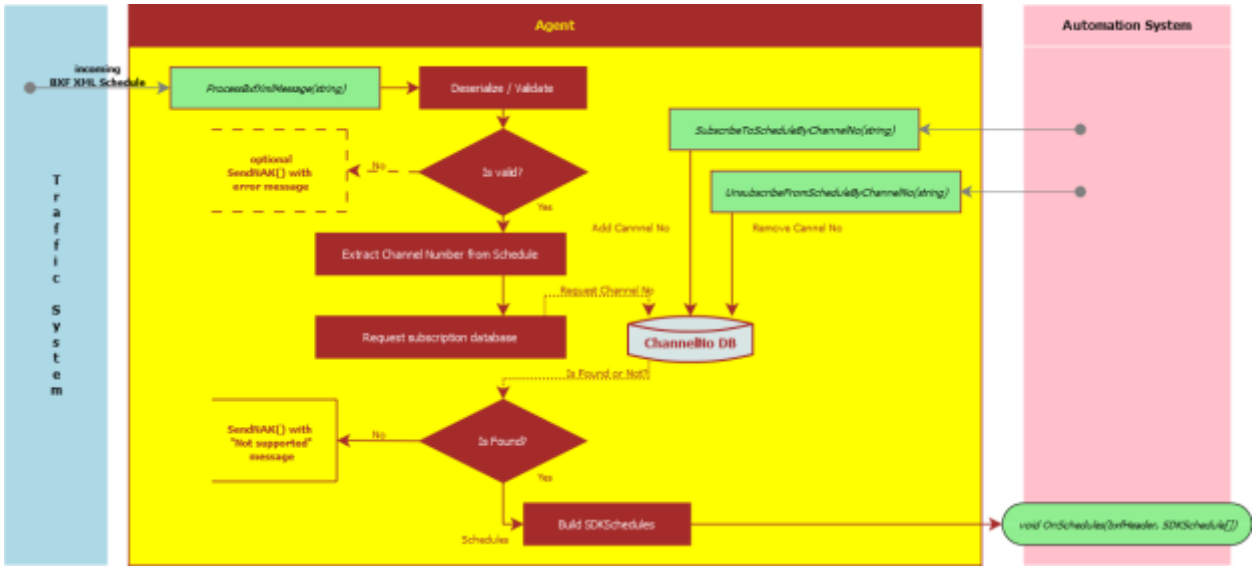


Fig. 1 Schedule subscription API

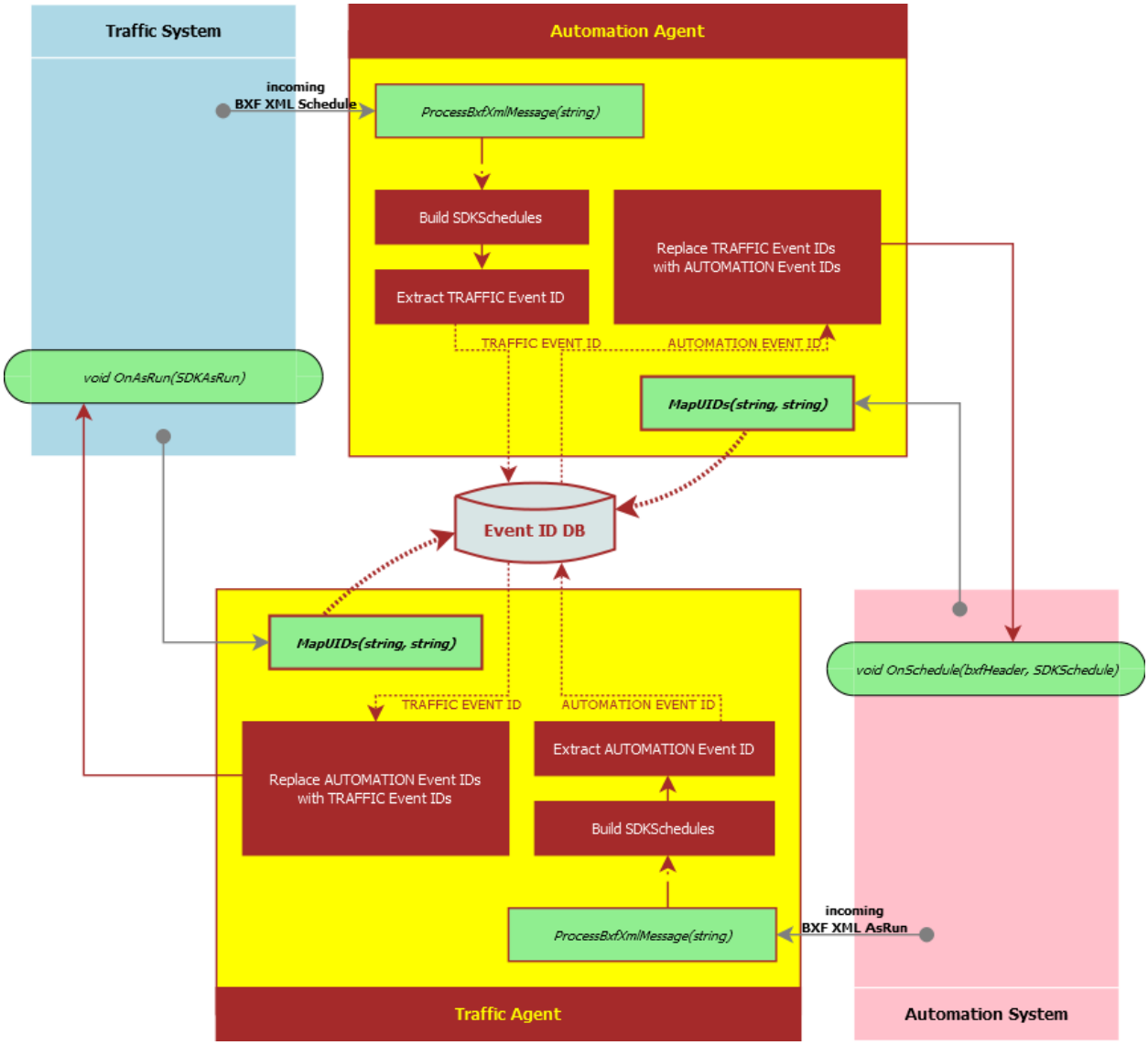


Fig. 2 Event ID mapping and replacement

If the Automation System and Traffic System generate their own UIDs for the same event ,then the SDK can substitute IDs in an agent module while processing a schedule and an asrun list. When an agent receives a schedule or an asrun list, it requests a special database (guidmap.db) for alternate event uid.

`MapUID (string trafficGuid, string automationGuid)` is a procedure to add a new pair of trafficId and automationId into the database for further usage.

5.5 Transport Adapter (HTTP Post Adapter)

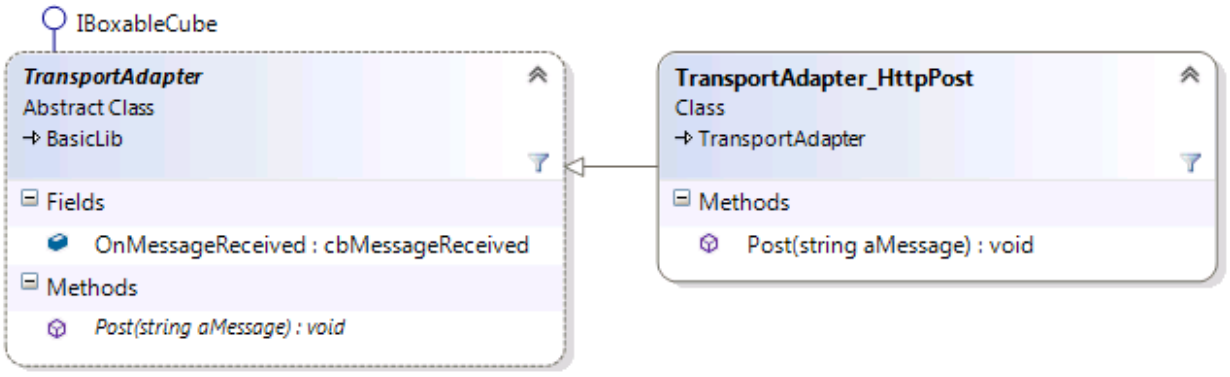


Fig. 3 TransportAdapter and TransportAdapter_HttpPost types

Event:	**OnMessageReceived**	**Occurs when the adapter receives a string message from a Partner**
Method:	Post(string aMessage)	Posts string message to a Partner. Returns nothing

5.6 BXF Validator

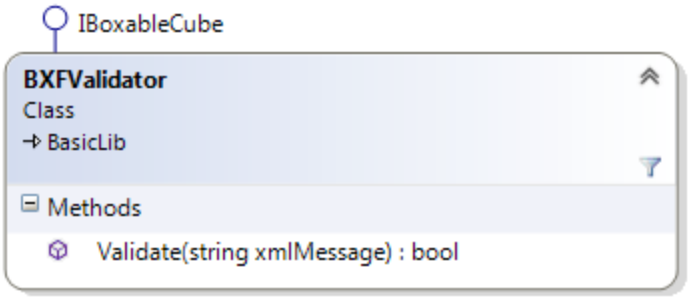


Fig. 4 BXFValidator type

Method:	**Validate(string xmlMessage)**	**Validates XML string message against to BXF XSD schema.** **See Configuration section.** **Returns:** ***true* - if the *xmlMessage* is valid xml document**

		***false* - otherwise**

5.7 Loggers

5.7.1 BXF Message Logger

BxfMessageLogger collects incoming (to SDK) and outgoing (from SDK) BXF messages to the Log/BxfMessages/YYYY-MM-DD folder (where YYYY-MM-DD is the date when a message has been processed).

Each message is saved as a file with a x-hh-mm-ss-nnn.xml name

Where:

x – prefix “in” for incoming messages and “out” for outcoming messages

hh-mm-ss-nnn – time when the message has been processed, nnn is msecs.

BXFMessageLogger is implemented as a singleton.

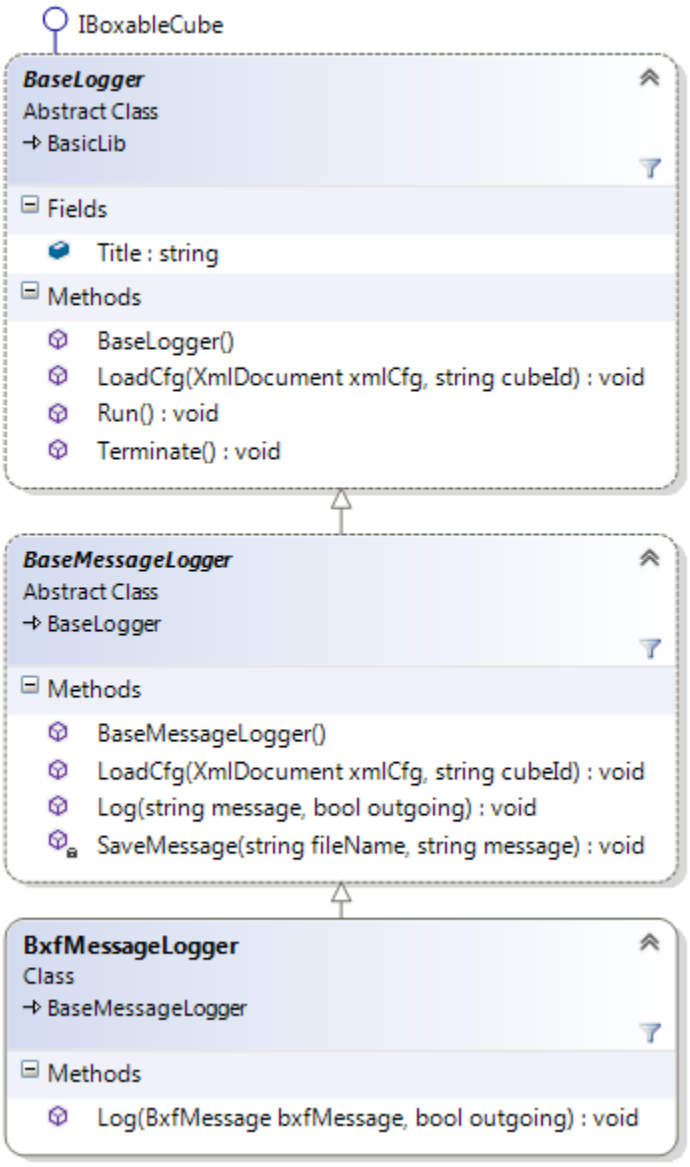


Fig. 5 BXFMessageLogger type

Method:	**LoadCfg(XmlDocument xmlCfg, string cubeId)**	**load logger configuration from *xmlCfg* object** ***cubeId* is to identify the logger in SDKBox**
	Run()	activate the logger
	Terminate()	deactivate the logger
	Log(BxfMessage bxfMessage,	Save *bxfMessage* ,

	bool outgoing)	if *outgoing* is *true* then save with “out” filename prefix if *outgoing* is *false* then save with “in” filename prefix

5.7.2 Error Logger

To simplify debugging and support, the SDK reports critical exception messages, warnings and debug info to the error log. The log is saved as an errorlog.html file in the working application directory. The file can be opened locally in any web-browser.

Any reported record contains 4 fields:

1) “Time” - date and time of the event or exception

2) “Thread” – a module where the event has occurred

3) “Level” – debug, warning or error

4) “Message” – event description with call stack for exceptions and source code detailed reference for errors

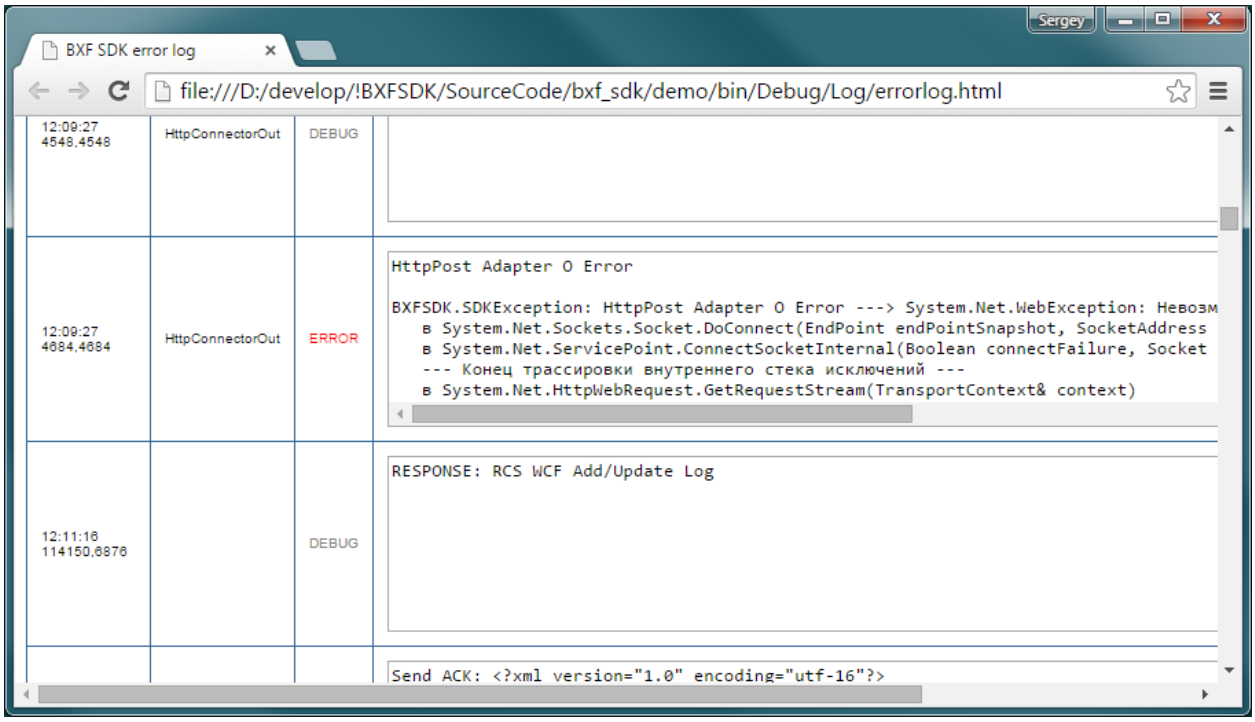


Fig. 6 Error log view

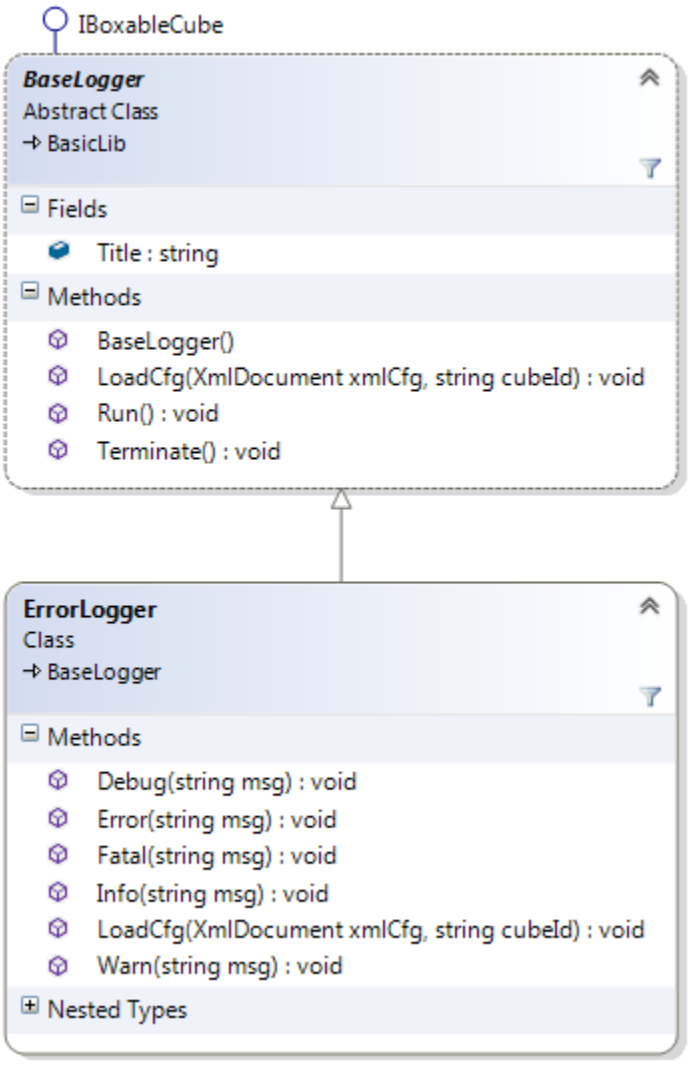


Fig. 7 ErrorLogger type

Method:	**LoadCfg(XmlDocument xmlCfg, string cubeId)**	**load logger configuration from *xmlCfg* object** ***cubeId* is to identify the logger in SDKBox**
	Run()	activate the logger
	Terminate()	deactivate the logger
	Info(string msg)	save the *msg* in the log with INFORMATION level marker
	Debug(string msg)	save the *msg* in the log with DEBUG level marker

	Warn(string msg)	save the *msg* in the log with WARNING level marker
	Error(string msg)	save the *msg* in the log with ERROR level marker
	Fatal(string msg)	save the *msg* in the log with FATAL level marker

5.8 Exception Handling

Exception handing in SDK cubes (see below) can be performed through their common box unit. Whenever any exception is raised in any cube thread, the SDK catches it and generates OnException event in SDKBox object. See SDKException type below.

5.1 SDKBox

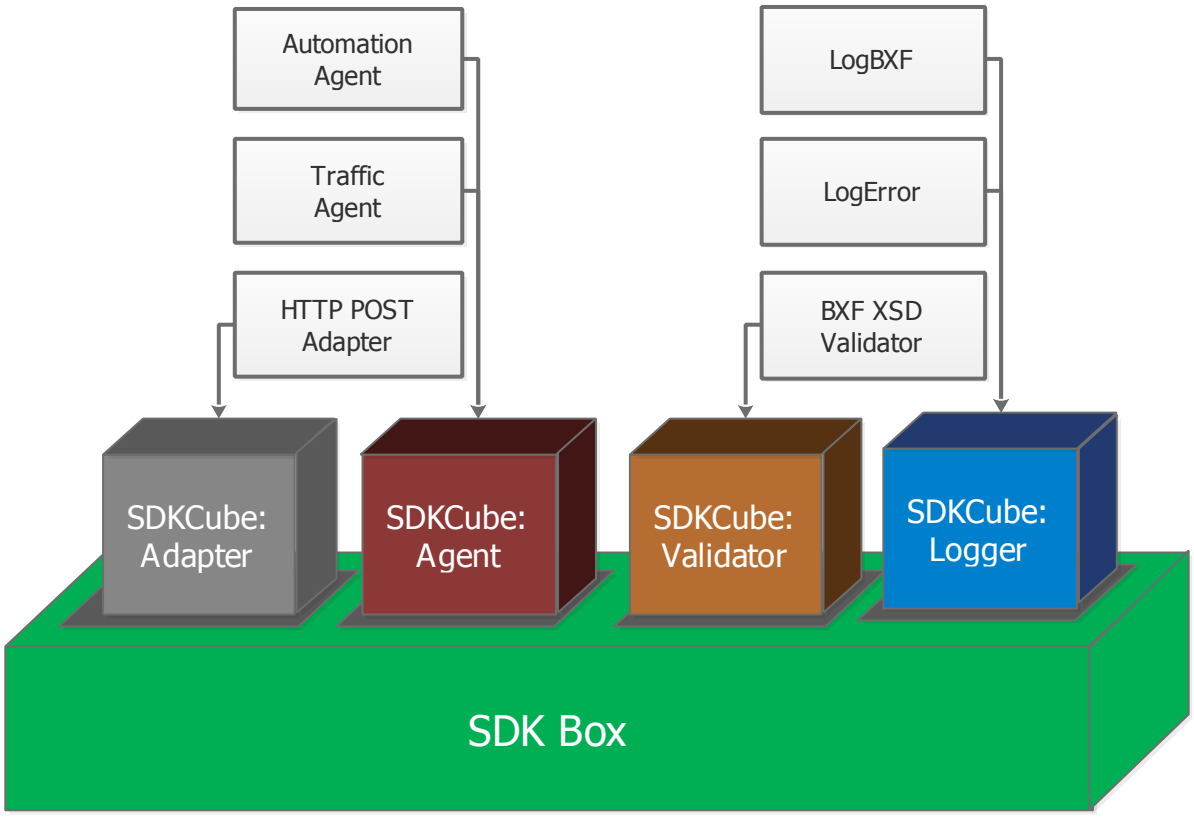


Fig. 8 SDKBox + SDKCubes design

SDK Box is a high level unit. Its purpose is to hold and handle lower level modules (cubes):

- Transport Adapters,

- Agents,

- Validator,

- Loggers.

This model allows using a few transport adapters of different types working simultaneously with the same agent by creating and inserting these adapters into the box.

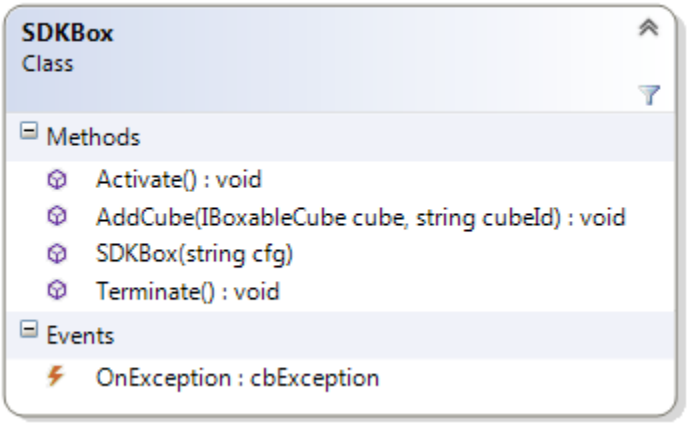


Fig. 9 SDKBox type

Event:	**OnException**	**Occurs when an exception is raised in any cube in the box.**
Methods:	SDKBox(string cfgFile)	Constructor. Creates SDKBox container object and loads SDK configuration settings from *cfgFile* file.
	AddCube(IBoxableCube cube, string cubeId)	Inserts a “cube” with cubeId into the box. cubeId value refers to cube section in a configuration file, it should be unique in the box.
	Activate()	Sets needed references between contained cubes and activates all cubes
	Terminate()	Finalize containing cubes activity

5.2 Assembly

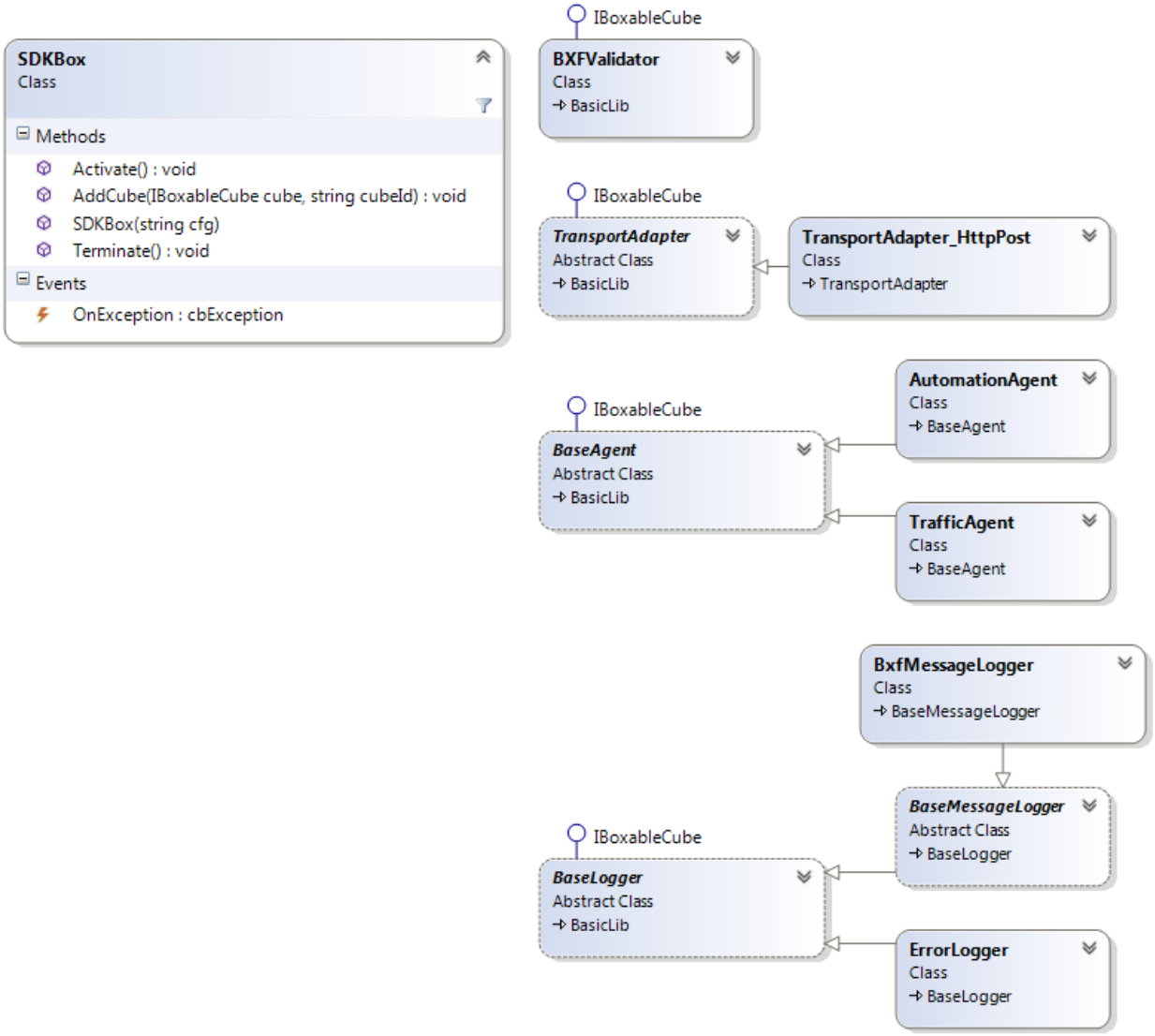


Fig. 10 SDKBox + Cubes types

On Traffic System side

```
BXFValidator xsdValidator = new BXFValidator();
SDKBox TrafficBox = new SDKBox(@"d:\conf.traffic.xml");
TrafficAgent TrafficAgent = TrafficAgent.Instance;
TransportAdapter_HttpPost TrafficAdapter = new TransportAdapter_HttpPost();
BxfMessageLogger BXFLog = BxfMessageLogger.Instance;
ErrorLogger ErrorLog = ErrorLogger.Instance;

TrafficBox.OnException += OnTrafficException;
```

```
TrafficBox.AddCube(TrafficAgent, "Agent");
TrafficBox.AddCube(TrafficAdapter, "HttpPostAdapter");
TrafficBox.AddCube(xsdValidator, "XsdValidator");
TrafficBox.AddCube(BXFLog, "BXFLogger");
TrafficBox.AddCube(ErrorLog, "ErrorLogger");

TrafficBox.Activate();

TrafficAgent.OnBxfXmlMessage += OnBxfXmlMessage;
...
bool OnBxfXmlMessage(string xmlString)
{
    //process xmlString
    ...
    return true;
}
void OnTrafficException(SDKException e)
{
    //process the exception
    ...
}
```

On Automation System side

```
BXFValidator xsdValidator = new BXFValidator();

SDKBox AutomationBox = new SDKBox(@"d:\conf.automation.xml");
AutomationAgent AutomationAgent = AutomationAgent.Instance;
TransportAdapter_HttpPost AutomationAdapter = new TransportAdapter_HttpPost();
BxfMessageLogger BXFLog = BxfMessageLogger.Instance;
ErrorLogger ErrorLog = ErrorLogger.Instance;

AutomationBox.OnException += OnAutomationException;

AutomationBox.AddCube(AutomationAgent, "Agent");
AutomationBox.AddCube(AutomationAdapter, "HttpPostAdapter");
AutomationBox.AddCube(xsdValidator, "XsdValidator");
AutomationBox.AddCube(BXFLog, "BXFLogger");
AutomationBox.AddCube(ErrorLog, "ErrorLogger");

AutomationBox.Activate();
AutomationAgent.OnBxfXmlMessage += OnBxfXmlMessage;
...
bool OnBxfXmlMessage(string xmlString)
{
    //process xmlString
    ...
    return true;
}
void OnAutomationException(SDKException e)
{
    //process the exception
    ...
```

}

5.3 Configuration

The SDKBox configuration file has an identical format for the Traffic side and for the Automation side.

It contains sections to define agent, validator and transport adapter(s) settings.

5.3.1 Agent Settings

<Agent…/>

userName, origin, originType and originId are inserted into outcoming BXF message headers. originId should be in “urn:uuid:-*GUID*-” format where -*GUID*- any valid GUID.

5.3.2 Validator Settings

<XsdValidator…/>

path to bxf xsd schema file and path to pmcp xsd schema file

5.3.3 HTTP POST transport adapter settings

<HttpPostAdapter…/>

listen – the adapter listens on this address

send – the adapter sends to this address

5.4 BXFMessageLogger

<LogBXF…/>

expireDays - determines the point at which message entries are expired and removed

incoming – true or false –should the logger record incoming messages or not

outcoming – true or false –should the logger record outcoming messages or not

5.5 Error Logger

<LogError…/>

info – true or false –should the logger record info level messages or not

debug – true or false –should the logger record debug level messages or not

warn – true or false – should the logger record warning level messages or not

error – true or false –should the logger record error level messages or not

fatal – true or false –should the logger record fatal error level messages or not

5.5.1 Examples

5.5.1.1 Automation configuration file example

```
<?xml version="1.0"?>
<Cubes>
<HttpPostAdapter
       listen       = "http://127.0.0.1:9902"
       send         = "http://127.0.0.1:9903" />
<Agent
       userName     = "BXF SDK Demo: automation"
       origin       = "Automation System"
       originType   = "Automation"
       originId     = "urn:uuid:-GUID-" />
<XsdValidator
       bxf = "./xsd/bxf_2008.xsd"
       pmcp = "./xsd/pmcp.xsd"/>
<LogBXF expireDays="3" incoming="true" outcoming="true"/>
<LogError debug="true" info="true" warn="true" error="true" fatal="true"/>
</Cubes>
```

5.5.1.2 Traffic configuration file example

```
<?xml version="1.0"?>
<Cubes>
<HttpPostAdapter
       send         = "http://127.0.0.1:9901"
       listen       = "http://127.0.0.1:9904" />
<Agent
       userName     = "BXF SDK Demo: traffic"
       origin       = "Traffic System"
       originType   = "Traffic"
       originId     = "urn:uuid: :-GUID-" />
<XsdValidator
       bxf = "./xsd/bxf_2008.xsd"
       pmcp = "./xsd/pmcp.xsd"/>
<LogBXF expireDays="3" incoming="true" outcoming="true"/>
<LogError debug="true" info="true" warn="true" error="true" fatal="true"/>
</Cubes>
```

5.6 User Interface Tool

In order to create configuration files for traffic or automation modules quickly and correctly, the BXF SDK Configurator application is provided. It allows the user to create a new file or to modify an existing one. Whenever all fields on “Agent”, “HttpPost Adapter” and “Xsd Validation File” tabs are filled in, the user must then define the agent type in the “Save As” field and click “Save”. The application overwrites the opened configuration file or creates a new file in current folder if the configuration file is not defined.
If Mule ESB Adapter or XSD validator is not added into box as a cube, it is possible to leave the adapter or the validator fields empty.

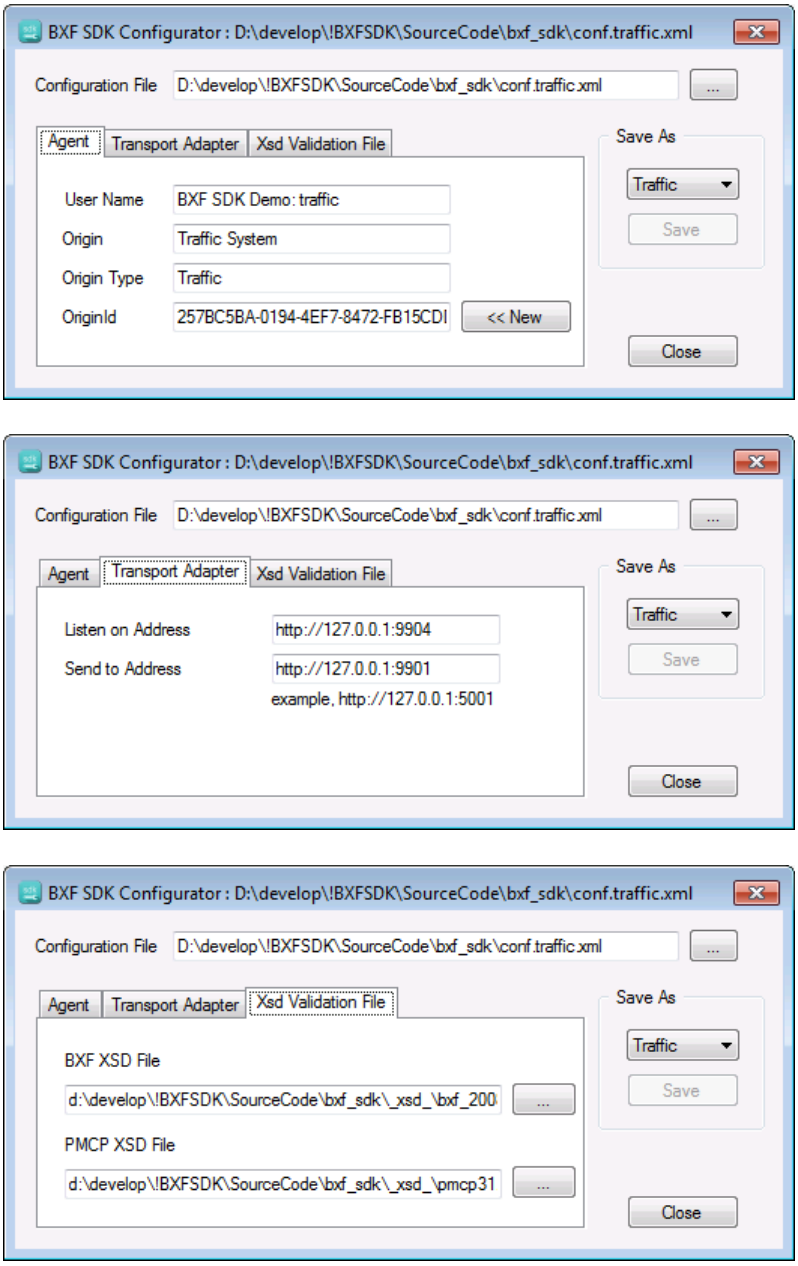


Fig. 11 Configurator UI

6 SDK Types

The following SDK types could be useful for simple descriptions and access to major TV automation units such as event and schedule. The original BXF format assumes a pretty sophisticated unit data structure with more extra fields which are not needed in most cases.

6.1 SDKEvent

SDKEventData describes a core event structure. SDKEvent and SDKAsRunEvent are inherited from SDKEventBase type.

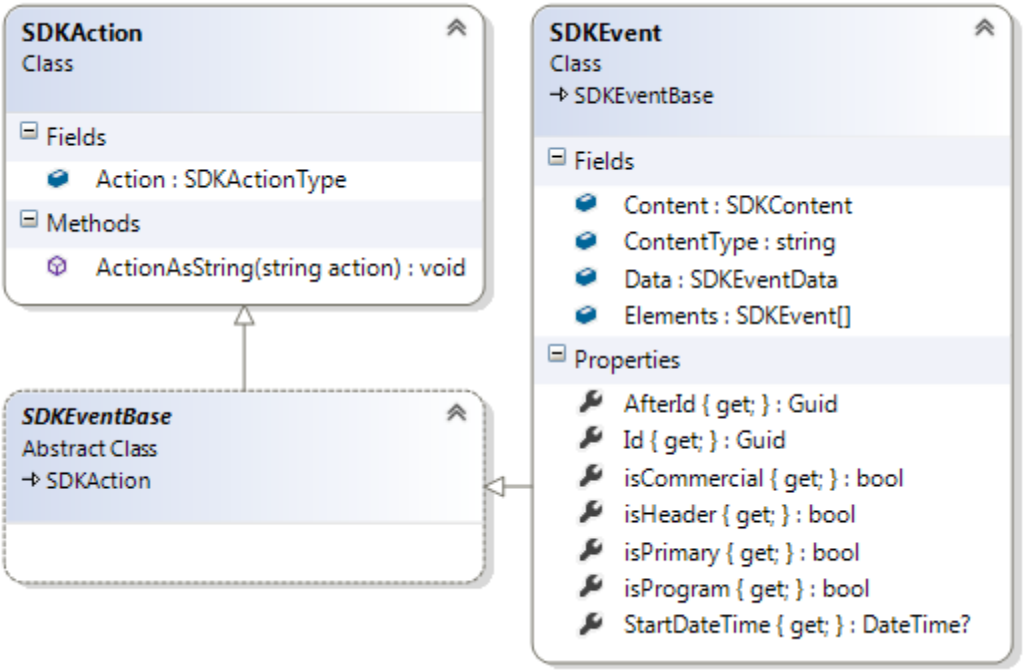


Fig. 12 SDKEvent type

Any event is transmitted in an SDKSchedule object where it needs Action, where Action is "*read*", "*add*", "*update*" or "*remove*".

Field/Property (Type)	Description	Relative BXF path from Schedule/ScheduledEvent
Action (SDKActionType)	describes what _action' is requested of the receiver of the message	`/@action`
Content (SDKContent)	includes video/audio content description as well as metadata for the content to be transferred from one location to another	`/Content`
ContentType (String)	categorizes content into specific groupings (e.g. network, local, news, entertainment)	`/ContentType`

Field/Property (Type)	Description	Relative BXF path from Schedule/ScheduledEvent
Data (SDKEventData)	Specifies the data for the complete show or a single event	/EventData or /ScheduleElements /EventData
Elements (SDKEvent[])	Specifies the elemental structure of the show	/ScheduleElements
Id (Guid)	Unique identifier for each event	/EventData /EventId /EventId
AfterId (GUID)	A pointer to the event prior to the position used to insert a new event in an existing list.	/EventData /InsertAfterId
isCommercial (bool)	is true for PrimaryBreakHeader or PrimaryNonProgram event	
isHeader (bool)	is true for PrimaryBreakHeader or PrimaryProgramHeader	
isPrimary (bool)	is true for PrimaryBreakHeader or PrimaryProgramHeader or PrimaryProgram or PrimaryNonProgram event	

Field/Property (Type)	Description	Relative BXF path from Schedule/ScheduledEvent
isProgram (bool)	is true for PrimaryProgram or PrimaryProgramHeader event	
Method	Description	
ActionAsString(String)	converts incoming value into SDKActionType	

6.2 SDKEventAsRun

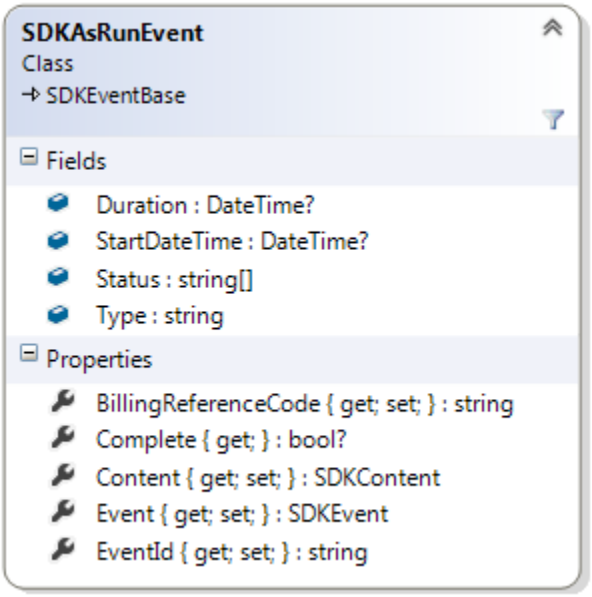


Fig. 13 SDKAsRunEvent type

The AsRun event type contains basic event definitions with asrun data. It is inherited from SDKEventBase. The BXF schema provides 2 types of asRun event definitions: short BasicAsRun type and extended CompleteAsRun. SDKAsRunEvent type holds the basic event definition as SDKEvent and presents as run data in both basic and complete forms.

Field/Property (Type)	Description	Relative BXF path from Schedule/AsRun
Duration (DateTime)	The actual duration of the event as aired.	/BasicAsRun /AsRunDetail /Duration or /CompleteAsRun /AsRunDetail /Duration
StartDateTime (DateTime)	The actual time the event started.	/BasicAsRun /AsRunDetail /StartDateTime or /CompleteAsRun /AsRunDetail /StartDateTime
Status (String[])	How the event was processed by the automation system	/BasicAsRun /AsRunDetail /Status or /CompleteAsRun /AsRunDetail /Status

Field/Property (Type)	Description	Relative BXF path from Schedule/AsRun
Type (String)	Enumerates the various types of asrun events: - Primary - NonPrimary - Auxillary - Comment - ProgramHeader - BreakHeader - Macro - Data	/BasicAsRun /AsRunDetail /Type or /CompleteAsRun /AsRunDetail /Type
Complete	is true if AsRun in CompleteAsRun form	
Actual for BasicAsRun form		
EventId (String)	References the scheduled event ID unless the event was added manually in which case this the Null value should be used.	/BasicAsRun /AsRunEventId
Content (SDKContent)	includes video/audio content description. Actual for BasicAsRun form only	/Content
Actual for CompleteAsRun form		

Field/Property (Type)	Description	Relative BXF path from Schedule/AsRun
BillingReferenceCode	Used to link an event back to its billing record.	/CompleteAsRun /EventData /EventId /BillingReferenceCode
Event (SDKEvent)	Specifies the data for the complete show or a single event.	/CompleteAsRun /EventData and /Content

6.3 SDKSchedule

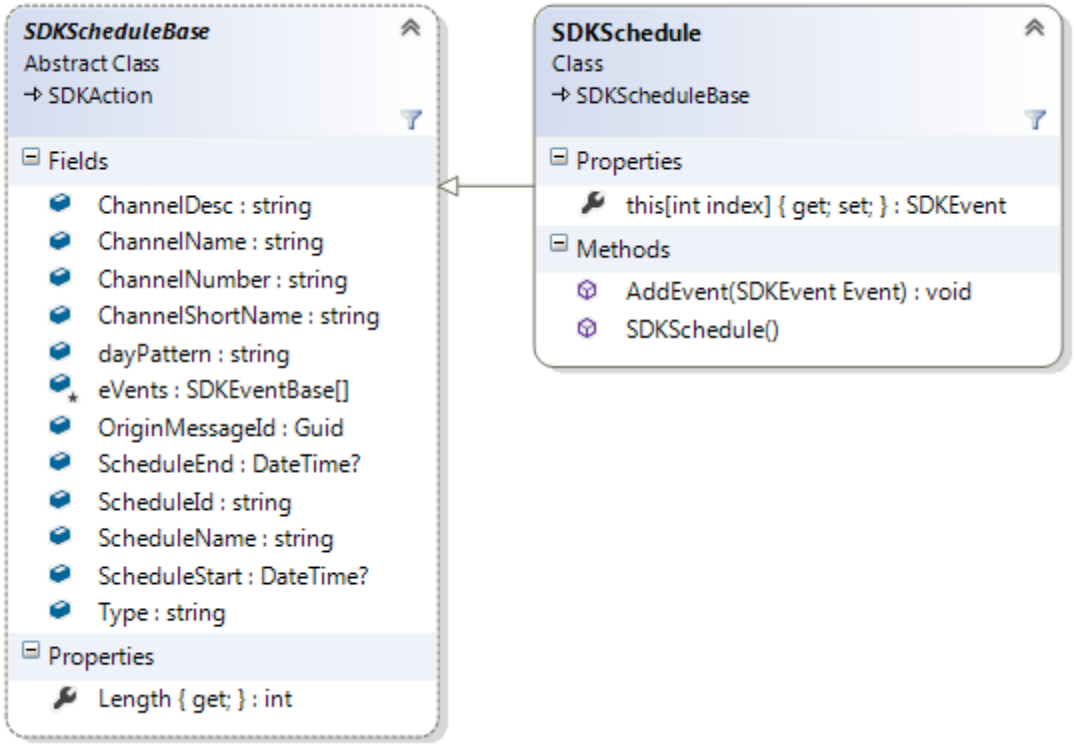


Fig. 14 SDKSchedule type

SDKSchedule is designed as a wrapper of BXF Schedule type.

Field/Property (Type)	Description	Relative BXF path from Schedule
ChannelDesc (string)	Multiple-language description of the channel carried in the channel ETT (A /65B 6.6)	`/Schedule /Channel /Description[0]`
ChannelName (string)	Long name of channel, reference to the first possible name from the set for multiple languages	`/Schedule /Channel /Name[0]`

Field/Property (Type)	Description	Relative BXF path from Schedule
ChannelNumber (string)	Two-part or one-part channel number of the virtual channel	/Schedule /Channel /@channelNumber
ChannelShortName (string)	PSIP short name (7 characters max.) (A/65B 6.3)	/Schedule /Channel /@shortName
dayPattern (String)	A binary representation of the days of the week with Monday in the left-most position	/Schedule /@dayPattern
OriginMessageId (Guid)	A BxfMessage UID. It is for response messages that refers to the message	BxfMessage /@id
ScheduleId (Guid)	A unique identifier for the schedule	/Schedule /@scheduleId
scheduleEnd (DateTime)	Date and Time of the End of the schedule	/Schedule /@scheduleEnd
ScheduleName (string)	A name to describe the schedule, reference to the first possible name from the set for multiple languages	/Schedule /ScheduleName[0]
scheduleStart (DateTime)	Date and Time of the start of a schedule	/Schedule /@scheduleStart

Field/Property (Type)	Description	Relative BXF path from Schedule
Length (int)	Count of events in the schedule	
operator this [int index]	Get access to scheduled event in the list by index	
Method	Description	
SDKSchedule()	constructor	
AddEvent(SDKEvent)	appends incoming event to the schedule	

6.4 SDKAsRunList

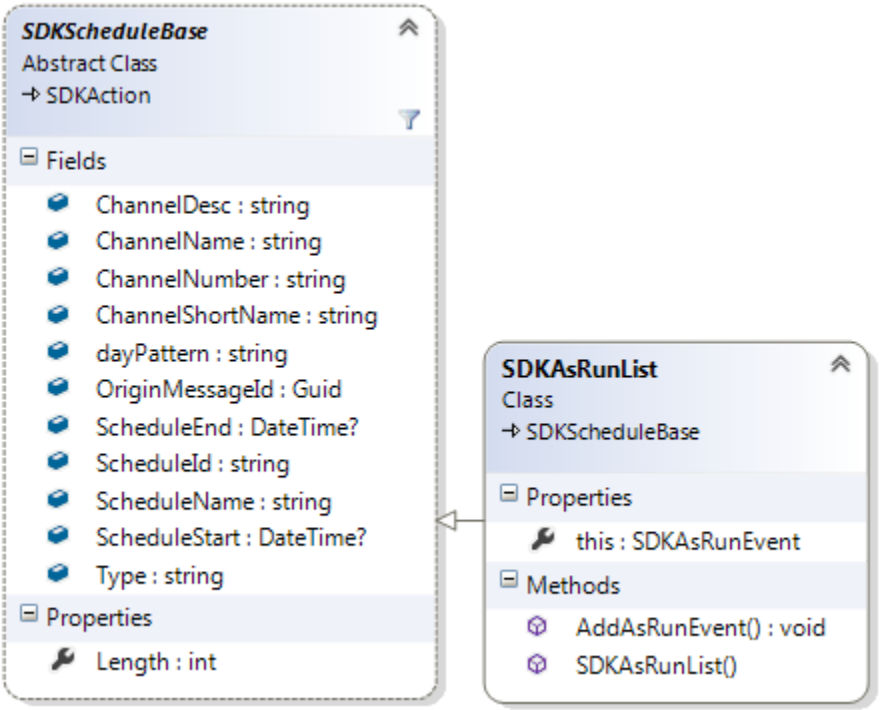


Fig. 15 SDKAsRunList type

Field/Property (Type)	Description	Relative BXF path from Schedule
ChannelDesc (string)	Multiple-language description of the channel carried in the channel ETT (A /65B 6.6)	`/Schedule /Channel /Description[0]`
ChannelName (string)	Long name of channel, reference to the first possible name from the set for multiple languages	`/Schedule /Channel /Name[0]`
ChannelNumber (string)	Two-part or one-part channel number of the virtual channel	`/Schedule /Channel /@channelNumber`

Field/Property (Type)	Description	Relative BXF path from Schedule
ChannelShortName (string)	PSIP short name (7 characters max.) (A/65B 6.3)	/Schedule /Channel /@shortName
dayPattern (String)	A binary representation of the days of the week with Monday in the left-most position	/Schedule /@dayPattern
OriginMessageId (Guid)	A BxfMessage UID. It is for response messages that refers to the message	BxfMessage /@id
ScheduleId (Guid)	A unique identifier for the schedule	/Schedule /@scheduleId
scheduleEnd (DateTime)	Date and Time of the End of the schedule	/Schedule /@scheduleEnd
ScheduleName (string)	A name to describe the schedule, reference to the first possible name from the set for multiple languages	/Schedule /ScheduleName[0]
scheduleStart (DateTime)	Date and Time of the start of a schedule	/Schedule /@scheduleStart
Length (int)	Count of events in the schedule	
operator this [int index]	Get access to asrun event in the list by index	

Field/Property (Type)	Description	Relative BXF path from Schedule
Method	Description	
SDKAsRunList()	constructor	
AddAsRunEvent(SDKEvent)	appends incoming event to the schedule	

6.5 SDKDubTransfer

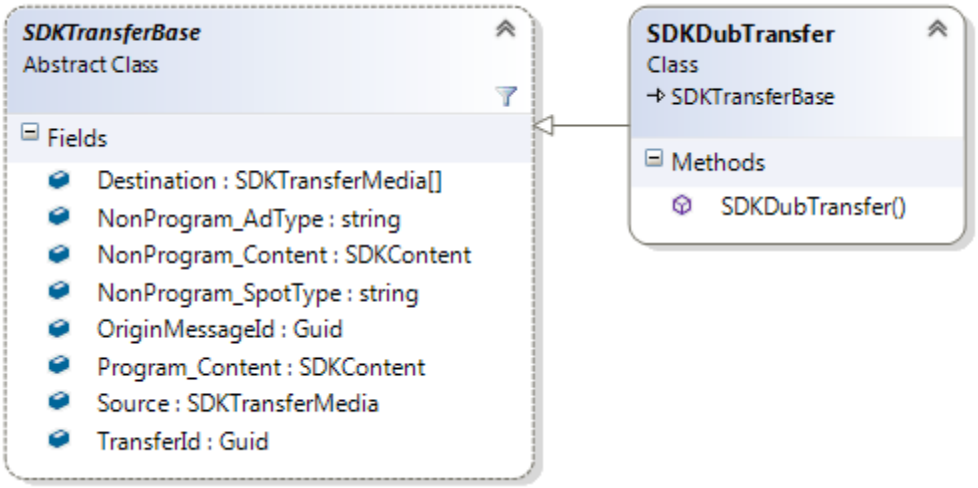


Fig. 16 SDKDubTransfer type

SDKDubTransfer defines an instruction that cause the specified content to be copied from its original location to a new destination. The type is inherited from SDKTransferBase.

Field/Property (Type)	Description	Relative BXF path from BxfData/ContentTransfer

OriginMessageId
 	
(Guid)
	A BxfMessage UID. It is for response messages that refers to the message	`BxfMessage`
		
		`/@id`

Field/Property (Type)	Description	Relative BXF path from BxfData/ContentTransfer
TransferId (Guid)	Identifies the transfer	/@transferId
Destination (SDKTransferMedia[])	A description of the location to be transfered to and optionally the transcoding of the content at this new location.	/Destination
Source	A description of the location and essence and any rules associated with its use	/Source
Program Content that is in the form of full length titles, epsiodes, movies, sports, newscasts, etc.		
Program_Content	Content identification and other details	/Content /ProgramContent /ContentMetaData
NonProgram Content that is typically short in nature and is scheduled in breaks between the segments of a program such as commercials, promos, ids, etc.		
NonProgram_AdType (String)	The type of advertisement being placed (e.g. General, Promo, PSA, etc.) BXF2013 ONLY	/Content /NonProgramContent /Details /AdType
NonProgram_Content (SDKContent)	Content identification and other details	/Content /NonProgramContent /ContentMetaData

Field/Property (Type)	Description	Relative BXF path from BxfData/ContentTransfer
NonProgram _SpotType (String)	The type of spot or other interstitial (e.g. Standard, BillBoard, Bookend.)	/Content /NonProgramContent /Details /SpotType
Method	Description	
SDKDubTransfer()	constructor	

6.6 SDKPurgeTransfer

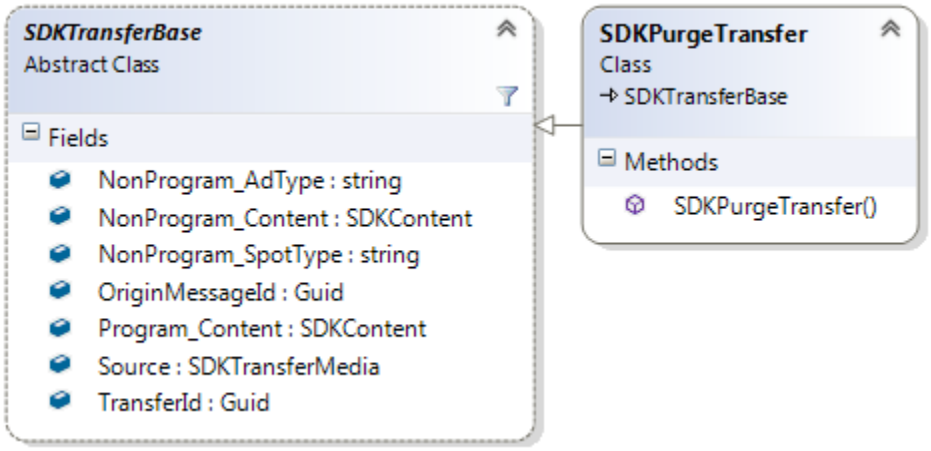


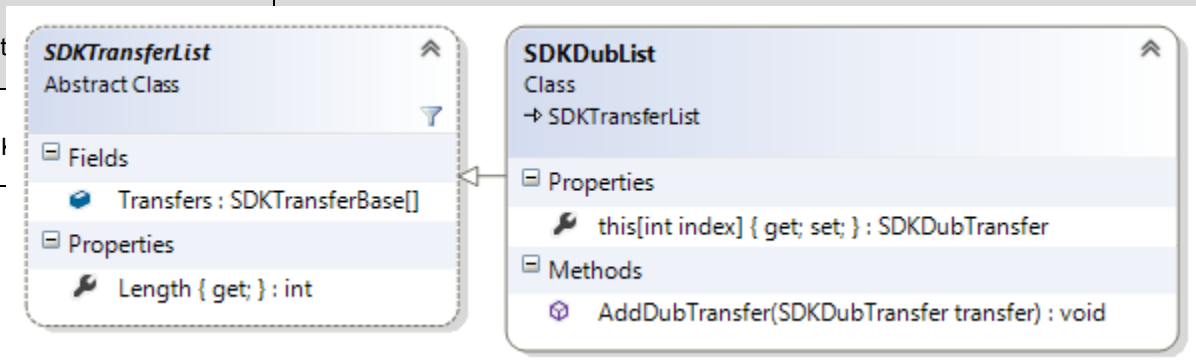
Fig. 17 SDKPurgeTransfer type

SDKPurgeTransfer defines an instruction that determines when a system should delete content from its current location. The type is inherited from SDKTransferBase.

Field/Property (Type)	Description	Relative BXF path from BxfData/ContentTransfer

Field/Property (Type)	Description	Relative BXF path from BxfData/ContentTransfer
OriginMessageId (Guid)	A BxfMessage UID. It is for response messages that refers to the message	BxfMessage /@id
TransferId (Guid)	Identifies the transfer	/@transferId
Source	A description of the location to be deleted from	/Source
Program Content that is in the form of full length titles, epsiodes, movies, sports, newscasts, etc.		
Program_Content	Content identification and other details	/Content /ProgramContent /ContentMetaData
NonProgram Content that is typically short in nature and is scheduled in breaks between the segments of a program such as commercials, promos, ids, etc.		
NonProgram_AdType (String)	The type of advertisement being placed (e.g. General, Promo, PSA, etc.) BXF2013 ONLY	/Content /NonProgramContent /Details /AdType
NonProgram_Content (SDKContent)	Content identification and other details	/Content /NonProgramContent /ContentMetaData

Field/Property (Type)	Description	Relative BXF path from BxfData/ContentTransfer
NonProgram _SpotType (String)	The type of spot or other interstitial (e.g. Standard, BillBoard, Bookend.)	/Content /NonProgramContent /Details /SpotType
Met		
SDI		



6.7

Fig. 18 SDKDubList type

SDKDubList defines a set of instructions that determines when a system should delete content from its current location. The type is inherited from SDKTransferList.

Field/Property (Type)	Description	Relative BXF path from BxfData
Transfers (SDKTransferBase[])	A set of transfer instructions	/ContentTransfer
Length (int)	Count of transfer instructions	
operator this[int index]	Get access to transfer instruction in the set by index	

Field/Property (Type)	Description	Relative BXF path from BxfData
Method	Description	
AddDubTransfer (SDKDubTransfer)	append new transfer instruction to the set	

6.8 SDKPurgeList

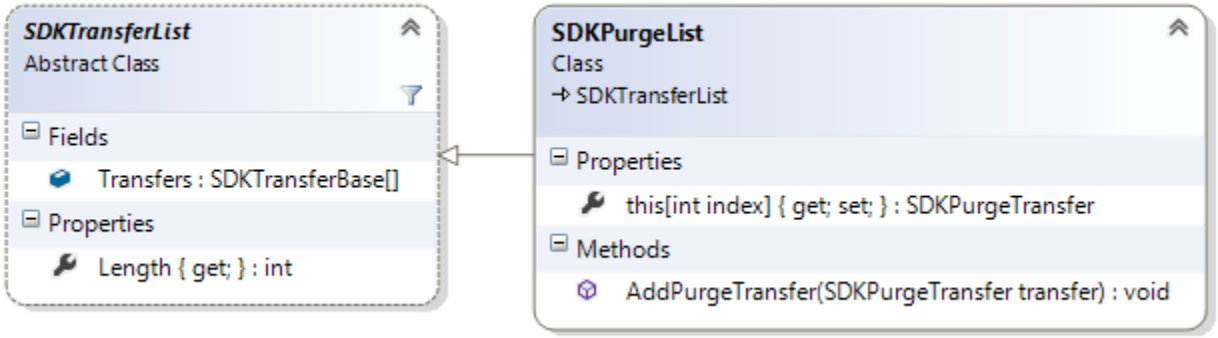


Fig. 19 SDKPurgeList type

SDKPurgeList defines a set of instructions that determine when a system should delete content from its current location. The type is inherited from SDKTransferList.

Field/Property (Type)	Description	Relative BXF path from BxfData
Transfers (SDKTransferBase[])	A set of transfer instructions	`/ContentTransfer`
Length (int)	Count of transfer instructions	
operator this[int index]	Get access to transfer instruction in the set by index	
Method	Description	
AddPurgeTransfer (SDKDubTransfer)	append new transfer instruction to the set	

6.9 SDKException

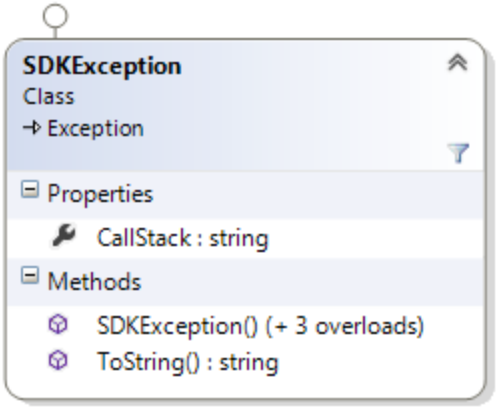


Fig. 20 SDKException type

Annex A SDK Internal Types (Informative)

SDKActionType

enums possible 'action' is requested of the receiver of the message. The action values are none (default undefined), read, add, update, remove.

	A set of values - none - read - add -update -remove	Enum	

SDKAction

describes what 'action' is requested of the receiver of the message. The action is made available at many types in the hierarchy in order to support precise meaning in messages that may be large and complex.

Field	BXF relative XPath	Description	Type BXF/SDK	
	/@action			
Action	@action	Read/ add/ update/ remove	Enum/ SDKActionType	N

SDKEventData

Basic event type containing core event description without related media data. SDKEventData item is included into SDKEvent type definition. API does not expect using an object of SDKEventData type directly.

Field	BXF relative XPath	Description	Type BXF/SDK	
	EventData/			
EventId	EventId/EventId or EventId/Null	Unique within a schedule reference for each event required to be a UUID or Null	Uuid/ String	?
Action	@action	Read/ add/ update/ remove	Enum/ String	N
BillingReferenceCode	EventId /BillingReferenceCode	Used to link an event back to its billing record	String	N
EventTitle	EventTitle	Title of the event	String	N
InsertAfterEventId	InsertAfterEventId	A pointer to the event prior to the position used to insert a new event in an existing list. Required to be a UUID	Uuid/ String	N
EventType	@EventType	Indicates the type of event to be described Enum: Primary Primary-ProgramHeader (ProgramEvent) Primary-BreakHeader (NonProgramEvent) NonPrimary Comment Macro	Enum/ String	N
PrimaryProgram_SegmentNumber	1 PrimaryEvent /ProgramEvent /SegmentNumber	The ordering of the segments within the show	String	Y
PrimaryProgram_ProgramName	1 PrimaryEvent /ProgramEvent /ProgramName	The name of the show	String	N
PrimaryNonProgram_AdType	2 PrimaryEvent /NonProgramEvent /Details /AdType	The type of advertisement (e.g. General, Promo, PSA, etc.) *BXF2013 only*	Enum/ String	Y
PrimaryNonProgram_SpotType	2 PrimaryEvent /NonProgramEvent	The type of spot or other interstitial (e.g. Standard, BillBoard, Bookend)	Enum/ String	Y

		/Details /SpotType			
NonPrimary_Name	3	NonPrimaryEvent /NonPrimaryEventName	Used to define the type of Secondary event (eg. Key, GPI, Tone, Audio or Video Effects)	String	Y
NonPrymary_PrimaryEventIdRef	3	NonPrimaryEvent /PrimaryEventId	Linkage to the primary event id. Not required for auxiliary events. Required to be a UUID.	Uuid	N
NonPrymary_Offset [0…2]	3	NonPrimaryEvent /Offset /OffsetTime /SmpteTimeCode	The time to start the Secondary event relative to the start time of the primary event	String[]	N
Macro_Name	4	MacroEvent /MacroName	Macro Event	String	Y
StartDateTime	StartDateTime /SmpteDateTime /@broadcastDate + StartDateTime /SmpteDateTime /SmpteTimeCode or StartDateTime /UtcDateTime		The date and time of the beginning of the event	DateTime	Y
EndDateTime	LengthOption /EndDateTime /SmpteDateTime /@broadcastDate + LengthOption /EndDateTime /SmpteDateTime /SmpteTimeCode or LengthOption /EndDateTime /UtcDateTime		End time of the event.	DateTime	Y
Duration	LengthOption /Duration /SmpteDuration /SmpteTimeCode or LengthOption /Duration /UtcDuration (as string)		Duration of the event.	DateTime	Y
StartMode	StartMode		The way the event will start relative to the previous event or time	Enum/ String	Y
EndMode	EndMode		The way an event will end relative to its duration or other events	Enum/ String	Y
AudioTransitionMode [0…]	AudioTransitions /AudioMode		Indicates whether the audio should go over, under or is a breakaway	Enum/ String	N
AudioTransitionType [0…]	AudioTransitions /TransitionType		Indicates if the transition is a cut or a mix (crossfade)	Enum/ String	N
AudioTransitionRate [0…]	AudioTransitions /TransitionRate		The speed of the transition: fast, medium or slow	Enum/ String	N
VideoTransitionPattern	VideoTransitions /TransitionPattern		If the transition is a wipe the pattern to be used for the wipe.	String	N
VideoTransitionInType	VideoTransitions /TransitionInType		The type of transition to perform between the previous event and this event	Enum/ String	N
VideoTransitionInRate	VideoTransitions /TransitionInRate		The rate of transition between the previous event and this event.	Enum/ String	N
VideoTransitionOutType	VideoTransitions /TransitionOutType		The type of transition to before between this event and the next event.		N
VideoTransitionOutRate	VideoTransitions /TransitionOutRate		The rate of transition between this event and the next event.	Enum/ String	N
VideoTransitionSom	VideoTransitions /TransitionSom		Used to designate a specific start point for the transition relative to the active content.	String	N

		/SmpteTimeCode			
VideoTransitionEom		VideoTransitions /TransitionEom /SmpteTimeCode	Used to designate a specific end point for the transition relative to the active content	String	N

SDKContent

Field	BXF relative XPath		Description	Type BXF/SDK	
	Content/	This is not expected to be present for comments, program headers and break headers, but is expected in all other cases.			
Action		@action	Read/ add/ update/ remove	Enum/ String	N
HouseNumber		ContentId /HouseNumber	Generally user defined at a specific site and will only be consistent within the defined group of users There is no guarantee of transportability or uniqueness.	String	Y
AlternateId		ContentId /AlternateId	Any proprietary string other than the house number used to identify the material locally or globally	String	N
AlternateIdType		ContentId /AlternateId @idType	This attribute uniquely identifies the type of alternate Id communicated in the element	String	N
Name		Name[0..]	Content Name	String[]	N
Description		Description[0…]	Content Description	String[]	N

SDKEvent

Field	BXF relative XPath		Description	Type BXF/SDK	
	Schedule/ScheduledEvent/ Schedule/ScheduledEvent/ScheduleElements/ CompleteAsRun/				
Data		EventData/	Specifies the data for the complete show or a single event	SDKEventData	Y
Content		Content/	This is not expected to be present for comments, program headers and break headers, but is expected in all other cases.	SDKEventContent	Y
ContentType	*)	ContentType/	Used to describe the type of content (eg. network, local, entertainment, news)	String	N
Elements	*)	SchedulesElements[]/	Specifies the elemental structure of the show	SDKEvent[]	N
Action	*)	@action	Read/ add/ update/ remove	Enum/ String	N
*) item is not specified for ScheduleElements					

SDKAsRunEvent

Field		BXF relative XPath		Description	Type BXF/SDK	
		AsRun/ CompleteAsRun/				
Complete asrun	Status		AsRunDetail /Status [1…]	How the event was processed by the automation system	Enum/ String[]	Y
	Type		AsRunDetail /Type	Enumerates the various types of asrun events.	Enum/ String	Y
	StartDateTime		AsRunDetail /StartDateTime /SmpteDateTime /@broadcastDate + AsRunDetail /StartDateTime /SmpteDateTime /SmpteTimeCode or AsRunDetail /StartDateTime	The actual time the event started.	DateTime	Y

		/SmpteDateTime /UtcDateTime			
	Duration	AsRunDetail /Duration /SmpteDuration /SmpteTimeCode or AsRunDetail /Duration /UtcDuration	The actual duration of the event as aired.	DateTime	Y
	Event	EventData + Content + ContentType + ScheduledElements	Specifies the data for the complete show or a single event	EventData/ SDKEvent	N
	AsRun/ BasicAsRun /				
Basic asrun	Status	AsRunDetail /Status [1…]	How the event was processed by the automation system	Enum/ String[]	Y
	Type	AsRunDetai l/Type	Enumerates the various types of asrun events.	Enum/ String	Y
	StartDateTime	AsRunDetail /StartDateTime /SmpteDateTime /@broadcastDate + AsRunDetail /StartDateTime /SmpteDateTime /SmpteTimeCode or AsRunDetail /StartDateTime /SmpteDateTime/UtcDateTim e	The actual time the event started.	DateTime	Y
	Duration	AsRunDetail /Duration /SmpteDuration /SmpteTimeCode or AsRunDetail /Duration /UtcDuration	The actual duration of the event as aired.	DateTime	Y
	EventId	AsRunEventId /EventId or AsRunEventId /Null	References the scheduled event ID unless the event was added manually in which case this the Null value should be used	Uuid/ String	Y
	BRC	AsRunEventId /BillingReferenceCode	Used to link an event back to its billing record	String	N
	Content	Content	Option specific reference to the content that aired used specifically when operator inserts content not originally scheduled. ! without ContentType	Content/ SDKContent	N

SDKSchedule

Field	BXF relative XPath	Description	Type BXF/SDK	
	Schedule/			
Action	@action	Read/ add/ update/ remove	Enum/ String	N

Type	@type	Primary or Alternate Schedule	Enum/ String {Primary}	Y
ScheduleId	@scheduleId	A unique identifier for the schedule	Uuid/ String	Y
DayPattern	@dayPattern	A binary representation of the days of the week with Monday in the left-most position (eg - "1111100" = M-F)	String	
ScheduleStart	@scheduleStart	Date and Time of the start of a schedule	DateTime?	N
ScheduleEnd	@scheduleEnd	Date and Time of the End of the schedule	DateTime?	N
ChannelNumber	Channel @channelNumber	Two-part or one-part channel number of the virtual channel	String	N
ChannelShortName	Channel @shortName	PSIP short name (7 characters max.) (A/65B 6.3)	String	N
ChannelName	Channel /Name[0..]	Long name of the channel, potentially for multiple languages (A/65B 6.9.5)	String[]	N
ChannelDesc	Channel /Description[0..]	Multiple-language description of the channel carried in the channel ETT (A/65B 6.6)	String[]	
ScheduleName	ScheduleName	A name to describe the schedule	String	N
Events	ScheduledEvent[1…]	Scheduled events list	SDKEvent[]	Y

SDKAsRunList

Field	BXF relative XPath	Description	Type BXF/SDK	
	Schedule/			
Type	@type	Primary or Alternate Schedule	Enum/ String {Primary}	Y
ScheduleId	@scheduleId	A unique identifier for the schedule	Uuid/String	Y
DayPattern	@dayPattern	A binary representation of the days of the week with Monday in the left-most position (eg - "1111100" = M-F)	String	
ScheduleStart	@scheduleStart	Date and Time of the start of a schedule	DateTime?	N
ScheduleEnd	@scheduleEnd	Date and Time of the End of the schedule	DateTime?	N
ChannelNumber	Channel @channelNumber	Two-part or one-part channel number of the virtual channel	String	N
ChannelShortName	Channel @shortName	PSIP short name (7 characters max.) (A/65B 6.3)	String	N
ChannelName	Channel /Name[0..]	Long name of the channel, potentially for multiple languages (A/65B 6.9.5)	String[]	N
ChannelDesc	Channel /Description[0..]	Multiple-language description of the channel carried in the channel ETT (A/65B 6.6)	String[]	
ScheduleName	ScheduleName	A name to describe the schedule	String	N
Events	AsRun[1…]	AsRun events list	SDKAsRunEvent[ ]	Y
Content				

SDKTransferMedia_BaseBand

Field	BXF relative XPath	Description	Type BXF/SDK	
	ContentTransfer/Source/Media/BaseBand/ or ContentTransfer/Destination/Media/BaseBand/	Used to describe the attributes of media that is still in the process of production prior to transmission or release for final viewing.		
xmlAudio	Audio		/XMLString	N
xmlVideo	Video		/XMLString	N
xmlCaption	Caption		/XMLString	N

SDKTransferMedia_PrecompressedTS

Field	BXF relative Xpath	Description	Type BXF/SDK	
	ContentTransfer/Source/Media/ PrecompressedTS/ or	Used to describe the attributes of the media that is to be transmitted or delivered for viewing as a complete package.		

	ContentTransfer/Destination/Media/ PrecompressedTS /				
xmlTSAudio	Audio		/XMLString	N	
xmlTSVideo	Video		/XMLString	N	
xmlTSData	Caption		/XMLString	N	

SDKTransferMedia_Location

Field	BXF relative Xpath	Description	Type BXF/SDK	
	ContentTransfer/Source/Media/ MediaLocation/ or ContentTransfer/Destination/Media/ MediaLocation /	Used to describe the attributes of the media that is to be transmitted or delivered for viewing as a complete package.		
Location	Location	Identification of where the media is stored. Used by systems to notify locations of essence instance	/XMLString	N
SOM	SmpteTimeCode	The start of message location of the content on the media it is stored.	/XMLString	N
Duration	Duration/ SmpteDuration/ SmpteTimeCode or Duration/ UtsDuration (as string)	The length of the content specified.	/XMLString	N

SDKTransferMedia

Field	BXF relative Xpath	Description	Type BXF/SDK	
	ContentTransfer/Source or ContentTransfer/Destination	A description of the location and essence and any rules associated with its use		
BaseBand	Media/ BaseBand	Used to describe the attributes of media that is still in the process of production prior to transmission or release for final viewing	/SDKMediaTransfer_BaseBand	N
PrecompressedTS	Media/ PrecompressedTS	Used to describe the attributes of the media that is to be transmitted or delivered for viewing as a complete package.	/SDKMediaTransfer_PrecompressedTS	N
Location[1..]	Media/MediaLocation or Media/MediaLocation[] BXF2013		/SDKMediaTransfer_Location[]	N

SDKDubTransfer / SDKPurgeTransfer

	Field	BXF relative XPath	Description	Type BXF/SDK	
		ContentTransfer/			
	TransferId	@transferId	Required to be a UUID	Uuid/ String	Y
	TransferType	@transferType	The purpose of the transfer order: Recording / Duplication / File transfer / Purge = "Duplication" ← For SDKDubTransfer = "Purge"← For SDKPurgeTransfer	Enum/ String	Y
1	Program_ Content	Content/ ProgramContent/ ContentMetaData	Content identification and other details		N
2	NonProgram_Content	Content/ NonProgramContent/ ContentMetaData	Content identifier and other information	ContentMetaData/ SDKContent	N
2	NonProgram_AdType	Content/ NonProgramContent/ Details/AdType	Used to describe the type of advertisement being placed (e.g. General, Promo, PSA, etc.) *BXF2013 ONLY*	Enum/ String {General}	N
2	NonProgram _SpotType	Content/	Used to describe the type of spot or other	Enum (BXF2013)	N

	NonProgramContent/ Details/ SpotType	interstitial (e.g. Standard, BillBoard, Bookend.)	or Value/ String {Standard}	
Source	ContentTransfer/ Source	A description of the location and essence and any rules associated with its use	/SDKTransferMedia	N
Destination[0..]	ContentTransfer/ Destination[]	A description of the location to be transfered to and optionally the transcoding of the content at this new location.	/SDKTransferMedia[]	N
OriginMessageId	BxfMessage/@id	A BxfMessage UID. It is for response messages that refers to the message	Guid	
SDKDubList / SDKPurgeList				
Transfers[0..]			SDKDubTransfer[] or SDKPurgePransfer[]	Y