

SMPTE RECOMMENDED PRACTICE

Implementing Broadcast Exchange Format (BXF)



Table of Contents	Page
Foreword	3
Intellectual Property	3
Introduction.....	3
1 Scope	4
2 Conformance Notation	4
3 Normative References	5
4 Note on Wellformedness and Validation (Informative)	5
5 Processing Model (Normative).....	5
6 General Conventions (Normative)	5
6.1 BxfMessage Attributes	6
6.2 Action Oriented Messages.....	7
6.3 Query Oriented Messages.....	7
6.4 Acknowledgment Messages.....	8
6.5 Reply Messages	8
6.6 Heartbeat Messages.....	9
7 Transport Protocol (Normative).....	10
7.1 Connection-Based Transport.....	10
7.2 File-Based Transport	11
8 Message Type Usages by Application Type (Informative)	12
8.1 Asset Management.....	12
8.2 Automation.....	13
8.3 Program Management.....	14
8.4 Traffic.....	15
9 Configuration.....	16
9.1 Overview (Informative).....	16
9.2 Recommended Conventions and Configuration Types (Normative).....	19
9.3 Requesting Configuration Information (Queries) (Normative).....	27
9.4 Requesting New Configuration Values be Added (Normative)	30
9.5 Notifying Others when Values or Mappings Change (Normative).....	31
10 Content.....	34
10.1 Overview (Informative).....	34
10.2 Requesting Content Information (Queries) (Normative).....	42
10.3 Notifying Others when Content is Added (Normative).....	45
10.4 Notifying Others when Content is Updated (Normative)	46
10.5 Notifying Others when Content is Removed (Normative).....	47

11 Content Transfer	48
11.1 Overview (Informative)	48
11.2 Content Transfer Status Query (Normative)	49
11.3 Dub Order (Normative)	49
11.4 Purge Order (Normative)	50
11.5 Record Order (Normative)	50
11.6 Transfer Order (Normative)	51
11.7 Content Transfer Status (Normative)	52
12 Format	52
12.1 Overview (Informative)	52
12.2 Requesting Format Information (Queries) (Normative)	55
12.3 Notifying Others when Formats are Added	57
12.4 Notifying Others when Formats are Updated	57
12.5 Notifying Others when Formats are Removed	58
13 Schedule	58
13.1 Overview (Informative)	58
13.2 Requesting Schedule Information (Queries) (Normative)	64
13.3 EPG Schedule Notifications (Normative)	65
13.4 Playlist Notifications (Normative)	66
13.5 As-run Notifications (Normative)	68
Annex A Supported Extensions	70
A.1 Overview	70
A.2 Usage Attribute	70
A.3 Filename Element	70
A.4 Root Element Extensions	70
A.5 Extension Schemas	70
Annex B Message Type Usages	71
B.1 Registered Message Type Usages	71
B.2 Information Message Syntax	74
B.3 Request and Reply Message Syntax	74
B.4 Query Message Syntax	74
B.5 Custom Message Type Usages	74
B.6 Application-Specific Message Type Usages	74
Annex C Error Handling	76
C.1 Overview	76
C.2 Error Types	76
C.3 Setting Errors Contextually	77
C.4 Identifying Errors and Warnings	77
C.5 Errors Implementation within BXF	78
Annex D Protocol Transport	80
D.1 Legend	80
D.2 Request Messagess	80
D.3 Information Messages	83
D.4 Heartbeat Messages	87
Bibliography	90

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.

SMPTE RP 2021-9 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 Recommended Practice. 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.

This practice describes implementation recommendations for applications that plan to support SMPTE 2021, commonly known as Broadcast Exchange Format (BXF) version 4.0. The BXF protocol is a sophisticated language that allows different application types (e.g., traffic, automation, etc.) to share information in a standardized format. The standard leaves many of the implementation details to the applications that are utilizing it. For example:

- Applications are only required to implement the aspects of the protocol that are relevant to the type of integration they are providing.
- Content may be identified through an ISAN, house number and/or 1-N alternate identifiers.

This "gray area" makes it possible for applications to implement BXF solutions that, while compliant with the standard, may not be compatible with other BXF implementations. This recommended practice strives to address this ambiguity by identifying common implementation recommendations that can be shared amongst applications that intend to utilize the BXF protocol to exchange information.

1 Scope

This recommended practice has the following objectives:

- Describe the types of messages that will be communicated amongst applications and their corresponding formats within BXF SMPTE 2021.
- Describe the aspects of the BXF protocol which are recommended that applications support to achieve maximum interoperability and also to identify those aspects of the protocol that are not recommended at this time.
- Describe the approaches for representing errors within BXF messages.
- Describe how XML schema extensions should be implemented to maximize interoperability between systems.
- Describe the following extensions:
 - Usage attribute extension
 - Filename element extension
 - Configuration extension

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, followed by 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 recommended practice. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this recommended practice are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

SMPTE ST 2021-1:2015, Broadcast Exchange Format (BXF) — Requirements and Informative Notes

RFC 4122, A Universally Unique Identifier (UUID) URN Namespace, IETF, 5 July 2005

4 Note on Wellformedness and Validation (Informative)

Two approaches for determining the integrity of an XML document are wellformedness and validation. According to W3C: *A "well-formed" XML document is defined as an XML document that has correct XML syntax. This means: XML documents must have a root element; XML elements must have a closing tag; XML tags are case sensitive; XML elements must be properly nested; XML attribute values must always be quoted. This should not be confused with a valid XML document, which is defined as a "well-formed" XML document which also conforms to the rules of a Document Type Definition (DTD) or an XML Schema (XSD), which W3C supports as an alternate to DTD.*

The XML schemas that comprise the BXF protocol utilize a "lax" processing model for extensions (i.e., PrivateInformation). Therefore, extensions that have an associated XML schema will validate when a message is processed, whereas extensions without schemas must be wellformed, but will not be validated.

5 Processing Model (Normative)

The processing model describes the rules for handling elements and attributes within BXF messages and extensions to those messages. The following rules shall apply to this practice:

- Ignore elements and attributes that are recognized, but not supported by an implementation. For example, if a house number and alternate identifiers are provided to uniquely identify content and a message consumer only recognizes house number, then the alternate IDs should be ignored, as opposed to rejecting the message as invalid.
- Ignore elements and attributes that are specified within the extensible areas of the protocol (i.e., PrivateInformation) and are not recognized by the message consumer.

6 General Conventions (Normative)

The following conventions shall apply to all BXF messages:

- All element and attribute extensions to the BXF protocol shall have associated XML schemas (and namespaces) so that they can be uniquely identified and resolved.
- It is recommended that systems should utilize a single, shared repository for BXF schemas and any extensions.
- Message consumers shall accept or reject a received message in its entirety. For example, if half of the events on a playlist message can be accepted and the other half cannot, the entire message should be rejected. This convention does not prohibit applications from accepting messages with warnings, where certain values, such as a description, are truncated due to field size restrictions. In this case, the entire message should still be accepted.

- Optional elements and attributes that are not covered within this document may be included within any message. These values may be consumed by applications that understand them, but they are not required to process a message.
- References to GUID in SMPTE ST 2021-1 and in this recommended practice shall be interpreted as UUID, as specified in RFC 4122.
- All date/time elements and attributes within a message shall be explicitly or implicitly represented using canonical UTC.
- Messages shall be processed in time sequence where message order is retained between a client and server.
- Message elements shall be processed in the order/sequence that they are specified.
- A message may contain multiple instances of the same element (i.e., same unique identifier). For example, a schedule event is added and updated within an application. The associated changes may be combined into a single event element with an action of add, or specified using two event elements, with the add element followed by the update element, where both elements have the same unique identifier. All elements shall be processed in the specified sequence.
- **BxfMessage/@ext:usage:** The usage extension attribute shall be included on every BXF message. It identifies the payload of the message in a contextual manner so the message consumer can process it in the proper context. Refer to Annex B for a list of recognized usages as well as conventions for creating additional usages. Note that usage values are not configuration types and are not mapped or configured at runtime.
- **BxfMessage/BxfData/PrivateInformation/ext:Filename:** The filename extension element is included in a BXF data message when the associated data is provided in a file form instead of the body of the message. This approach may be utilized when the amount of data is exceedingly large and chunking is not desired.
- Message length shall not exceed 2 MB. Applications should use internal algorithms to separate data into multiple messages. For example, if a user selects a month's worth of schedule information to send, then the application should break the schedule information into multiple messages based on the date or some other criteria.

6.1 BxfMessage Attributes

- **BxfMessage/@id:** The id attribute is required by the BXF schema.
- **BxfMessage/@dateTime:** The date/time attribute is required by the BXF schema. It identifies when the message was created.
- **BxfMessage/@messageType:** The message type attribute is required by the BXF schema. It identifies the message type/category (e.g., request, reply, information, etc.).
- **BxfMessage/@origin:** The origin attribute is required by the BXF schema. It identifies the message originator. This value shall be defined at runtime as an aspect of configuration.
- **BxfMessage/@originType:** The origin type attribute is required by the BXF schema. It identifies the originator's type (e.g., Traffic system). If a valid type value does not exist in the ATSC code point registry, then a value of "Data_server" shall be used by default.
- **BxfMessage/@userName:** The user name attribute is required by the BXF schema. It identifies the user that made the application changes which resulted in the message being created. If the message was generated by a background process or service, then the process or service name shall be specified. For acknowledgement and reply messages that are typically generated automatically, the username that is specified on the original message shall be utilized unless the original message required manual processing by a different user.

- **BxfMessage/@status:** Per the BXF protocol documentation, the status attribute shall be provided on acknowledgement and reply messages.
- **BxfMessage/@error:** The error attribute shall be included if a message is rejected by a receiving system.
- **BxfMessage/@errorDescription:** The error description attribute shall be included if the message is rejected by a receiving system. These conventions are described later in this document.

All other attributes on the BxfMessage element are not required and may not need to be specified.

6.2 Action Oriented Messages

The BXF protocol allows message creators to specify the action (or actions) that are associated with a message. The action attribute exists on most elements within the protocol.

Use of action attributes should be limited to the **BxfMessage/BxfData** element structure, where the message type is either "Request" or "Information".

The overarching action for a message shall be specified so that message consumers will have a common location for identifying the action. The overarching action should be specified on the sub-elements of the BxfData element. Examples:

- **BxfMessage/BxfData/Content/@action**
- **BxfMessage/BxfData/Format/@action**
- **BxfMessage/BxfData/Schedule/@action**

This provides the capability for a single message to have multiple sub-elements with different actions. Such as a content message with multiple Content elements with add, update and remove actions.

If there is only one sub-element, or all of the sub-elements have the same action, then the action may be specified on the BxfData element:

- **BxfMessage/BxfData/@action**

Additional actions may be specified within the message as described within the BXF protocol documentation.

6.3 Query Oriented Messages

Query messages shall be utilized for the following purposes:

- To retrieve information from another application during initial setup/configuration, as well as to synchronize the data within applications when an out-of-sync condition occurs. For example, a traffic system queries a program management system (or vice-versa) to obtain Formats so that the format information within the applications can be synchronized.
- To retrieve detailed information, or information that is required but not stored in an application. For example, a traffic system queries an automation system to obtain timing information on program content which is not stored in the traffic system.

Conventions – Query Messages

- **BxfMessage/BxfQuery/@requestChunking:** The request chunking attribute determines if chunking is desired. Chunking may result regardless of whether it was requested.

- **BxfMessage/BxfQuery/@memoryLimit:** The memory limit attribute identifies the desired maximum message size on a query reply
- **BxfMessage/BxfQueryResponse/@chunkingFlag:** The chunking flag attribute determines if a query reply is chunked.
- **BxfMessage/BxfQueryResponse/@itemNumber:** The item number attribute identifies the number of the current chunk.
- **BxfMessage/BxfQueryResponse/@totalItems:** The total items attribute identifies the final chunk. When total items attribute equals the item number attribute it is final chunk. Note that the total items attribute may be provided on every chunk or only on the final chunk.
- **BxfMessage/BxfQueryResponse/ext:Filename:** The filename extension element is included in a BXF query response message when the requested data is provided in a file form instead of the body of the reply. This approach may be utilized when the response is exceedingly large and chunking is not desired.

6.4 Acknowledgment Messages

Per the BXF protocol documentation, applications that support asynchronous messaging in a connected environment shall acknowledge all information, request and heartbeat messages that are received.

Example messages:

[Successful Acknowledgement](#)

[Warning Acknowledgement](#)

[Error Acknowledgement](#)

[Error Acknowledgement \(malformed XML\)](#)

Conventions – Acknowledgment Messages

BxfMessage/@ext:usage: The usage shall be “Application Acknowledgment”.

BxfMessage/@messageType: The message type shall be “Acknowledgement”.

BxfMessage/@destination: Acknowledgement messages shall include a destination, which is the origin in the request message.

BxfMessage/@status: Acknowledgement messages shall include a status indicating if the original message was accepted or rejected.

BxfMessage/@action: Acknowledgement messages should not contain an action.

BxfMessage/@error: Acknowledgement messages shall include error(s) if the original message was rejected. Refer to examples above and the annex for additional information.

BxfMessage/@errorDescription: Acknowledgement messages should include an error description if the original message was rejected. Refer to examples above and the annex for additional information.

The acknowledgement shall precede the reply if the original message was a request.

6.5 Reply Messages

Per the BXF protocol documentation, applications that support asynchronous messaging in a connected environment must reply to all request messages that are received. Refer to remaining sections for example messages.

Conventions – Reply Messages

Replies shall only be generated in response to messages with a `BxfMessage@messageType` of "Request".

BxfMessage/@ext:usage: The usage is message specific. Refer to the reply sections of the document.

BxfMessage/@messageType: The message type shall be "Reply".

BxfMessage/@destination: Reply messages shall include a destination, which is the origin in the request message.

BxfMessage/@status: Reply messages shall include a status, which indicates if the request was processed successfully or not.

BxfMessage/@action: Reply messages should not contain actions.

BxfMessage/@error: Reply messages shall contain errors if the request was rejected.

BxfMessage/@errorDescription: Reply messages should include an error description if the original message was rejected.

6.6 Heartbeat Messages

Applications that support asynchronous messaging in a connected environment may use heartbeat messages to verify that another application is operational. Heartbeat capabilities are not required by the BFX protocol. It is important to note that regardless of whether an application is a sender of heartbeat messages, all applications are required by BFX (SMPTE ST 2021-1, Section 6.3.1) to acknowledge heartbeat messages.

6.6.1 Monitoring Other Applications

Applications may provide the capability to monitor other applications by creating and sending heartbeat messages. BFX describes additional capabilities that should be implemented with this functionality, such as a configurable interval for sending heartbeats. This functionality is not required by the BFX protocol.

Example messages:

Heartbeat
Heartbeat (targeted)

Conventions – Heartbeat Messages

BxfMessage/@ext:usage: The usage shall be "Application Heartbeat".

BxfMessage@messageType: The message type shall be "Heartbeat".

BxfMessage@destination: Heartbeat messages may be targeted to a single destination or sent to all destinations. The message originator is responsible for tracking application status by destination. Heartbeats with a destination attribute are targeted, while heartbeats with no destination are sent to all applications.

6.6.2 Acknowledging Heartbeat Messages

Applications shall provide the capability to acknowledge heartbeat messages. Heartbeat message acknowledgements are no different than any other type of acknowledgement in BFX.

Conventions – Heartbeat Acknowledgement Messages

BxfMessage/@ext:usage: The usage shall be "Application Acknowledgment".

BxfMessage@status: Heartbeat acknowledgements shall include a status of "OK". Error statuses are not applicable on heartbeat messages, since there is nothing to reject.

7 Transport Protocol (Normative)

7.1 Connection-Based Transport

The following sections describe connection oriented behaviors associated with BXF Messaging.

It is anticipated that prevalent technologies, such as web services, will be utilized within the transport layer. However, applications are free to utilize whatever technologies they desire within their API. Examples include, but are not limited to web services, HTTP Post, sockets and stored procedures.

- Web services should be used in situations requiring near real-time performance and where systems are loosely coupled, because they follow standardized protocols and procedures and can be language agnostic. They provide excellent performance and easily address accessibility issues including firewall access. Web services may not be appropriate for sending large messages, e.g. queries to retrieve all content metadata from another system.
- Sockets may also be used for situations requiring near real-time performance and where systems are loosely coupled and are situated on the same network. They may be used in other situations providing appropriate ports are exposed to connecting systems.
- Stored procedures may be used in cases involving tightly coupled integrations, i.e. in situations where the database is from the same vendor. Stored procedures provide extremely good performance, but require systems to be on the same network or need an intermediary to cross networks, as well as access to the server to communicate with an application database.
- File-based connections may be used for sending very large messages or in cases where near real-time performance is not a requirement. Either network file or FTP may be used as a delivery method. File-based connections may be useful for delivering very large queries or larger messages such as configuration values. Please note that as an alternative to file-based delivery for large files, message chunking may be used.

Applications have the flexibility to return an acknowledgement message as a return value or through a separate call to the server that the message originated from. The implementation approach depends on the type of application, the type of information and other factors, such as performance. Refer to Annex D, Protocol Transport, for additional information.

7.1.1 Default Ports

Per the BXF protocol, the default ports are 14544 and 14545. This does not preclude the use of other ports, such as port 80, for web services.

7.1.2 Loss of Communications

At any point in time, messages may be interrupted due to loss of connectivity. This loss may occur for any of a numbers of reasons, and may occur at any time.

As defined in the standard, Section 7.1.6 covers how to handle timeouts in communications. Anytime this situation occurs, the sending system can enter into a retry loop to reestablish communications. In addition, the receiving system will also enter into a retry loop.

During this time of loss of communications, no messaging can occur. Each affected system is responsible for buffering messages until such time as the systems are returned to normal communications.

7.1.3 Validating Sources and Destinations

Sources and Destinations shall be identified and configured prior to communication via BXF may occur. Once these names are configured, then systems may recognize messages from other systems.

There are two scenarios that can occur prior to proper configuration of these names, and are important to the integrity of operations between systems. These are:

- In the case of a message received from an unrecognized source, the receiving system may not make any reply. This will prevent external systems from “phishing” to attempt to breach security within the environment.
- In the case of a message received from a recognized source which incorrectly identifies the destination, such as the mismatch between the names of identified systems vs. the name as configured in the receiving system, the receiving system may NAK the message.

Systems that receive messages that do not conform to these two cases may choose to log these occurrences. This may be helpful when troubleshooting communications when configuring the systems initially.

7.1.4 Server-Only Applications

Server-only applications provide services for other applications to call, but do not have client functionality necessary to call services provided by other applications. Server-only applications shall acknowledge messages using the return value approach specified below. Server-only applications may be notified of changes to data (i.e., information messages) but cannot support a request for information since the corresponding reply message must be provided on a separate call to the originating application's server.

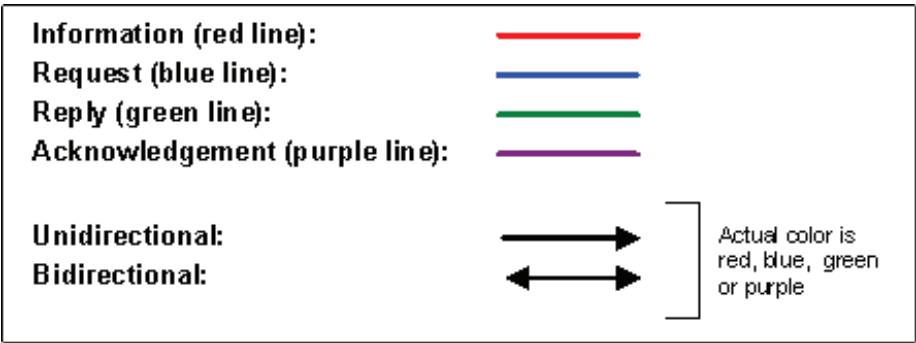
7.2 File-Based Transport

The basic messaging constructs used for connection-based transport are equally applicable to messaging using files. The only change that may be needed is the adjustment of the timeout configuration. This will allow implementations to use either a polling method or a notification method for processing of messages.

Note that there are cases where file-based operations will be necessary based on the size of returned messages. This implies that anytime connection based operations are implemented, file based may also be used.

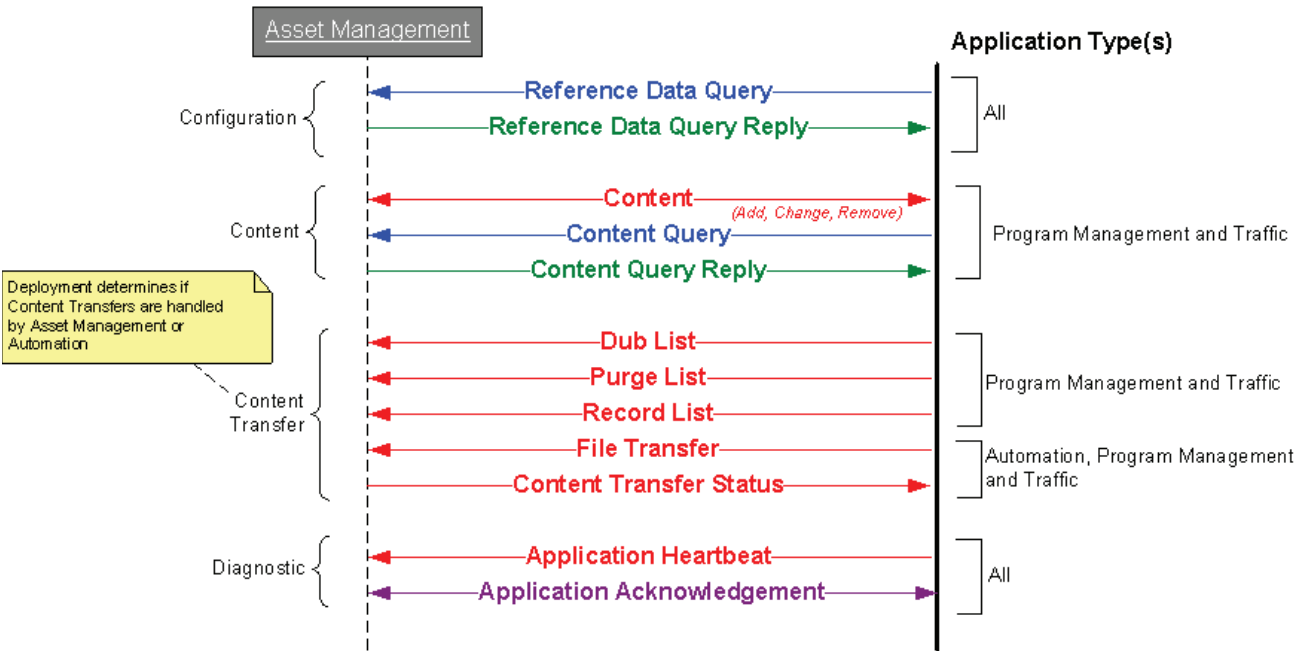
8 Message Type Usages by Application Type (Informative)

The following section identifies major communication links between applications based on their type and the type of information being sent (i.e., message type usage). Note that the message type usages in these sections are a subset of the possible usages defined in Annex B, Message Type Usages. Applications can implement as many usages as desired, based on their functionality.



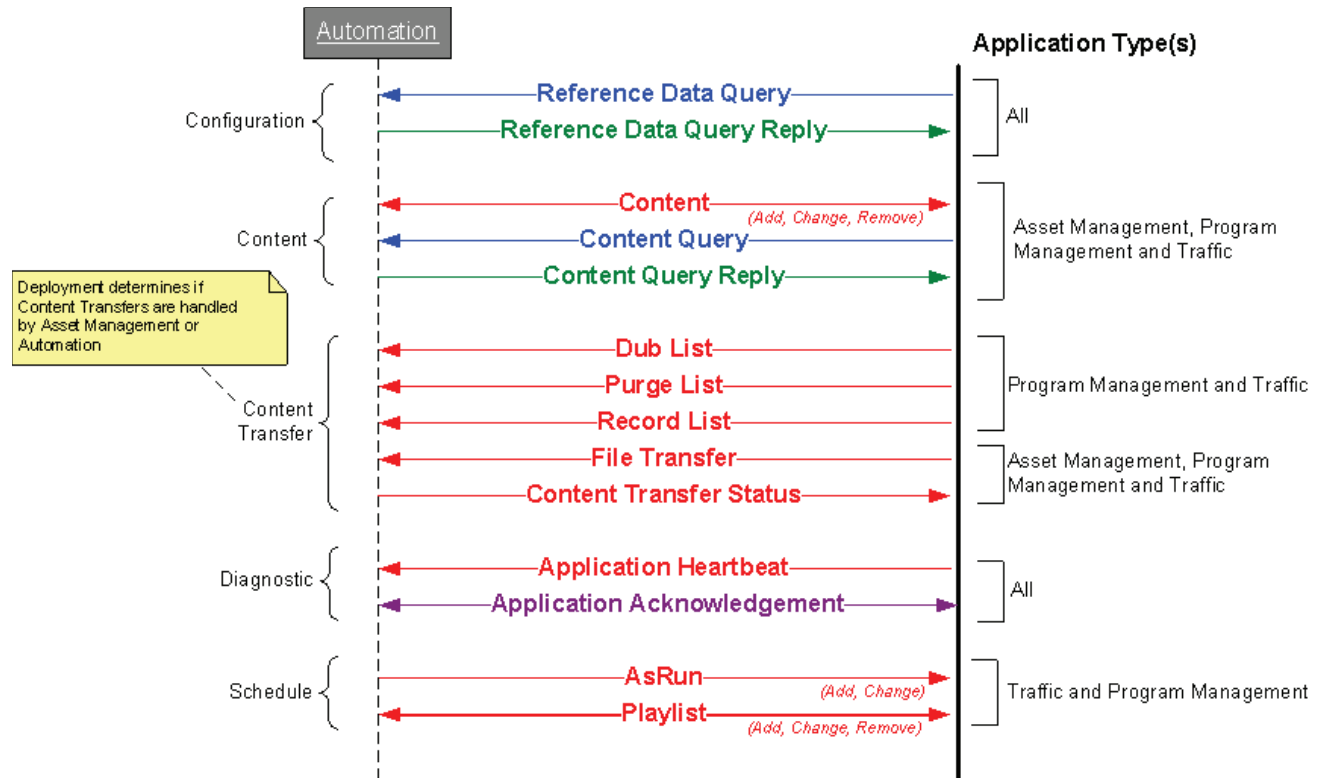
8.1 Asset Management

The following message usages are recommended for asset management applications.



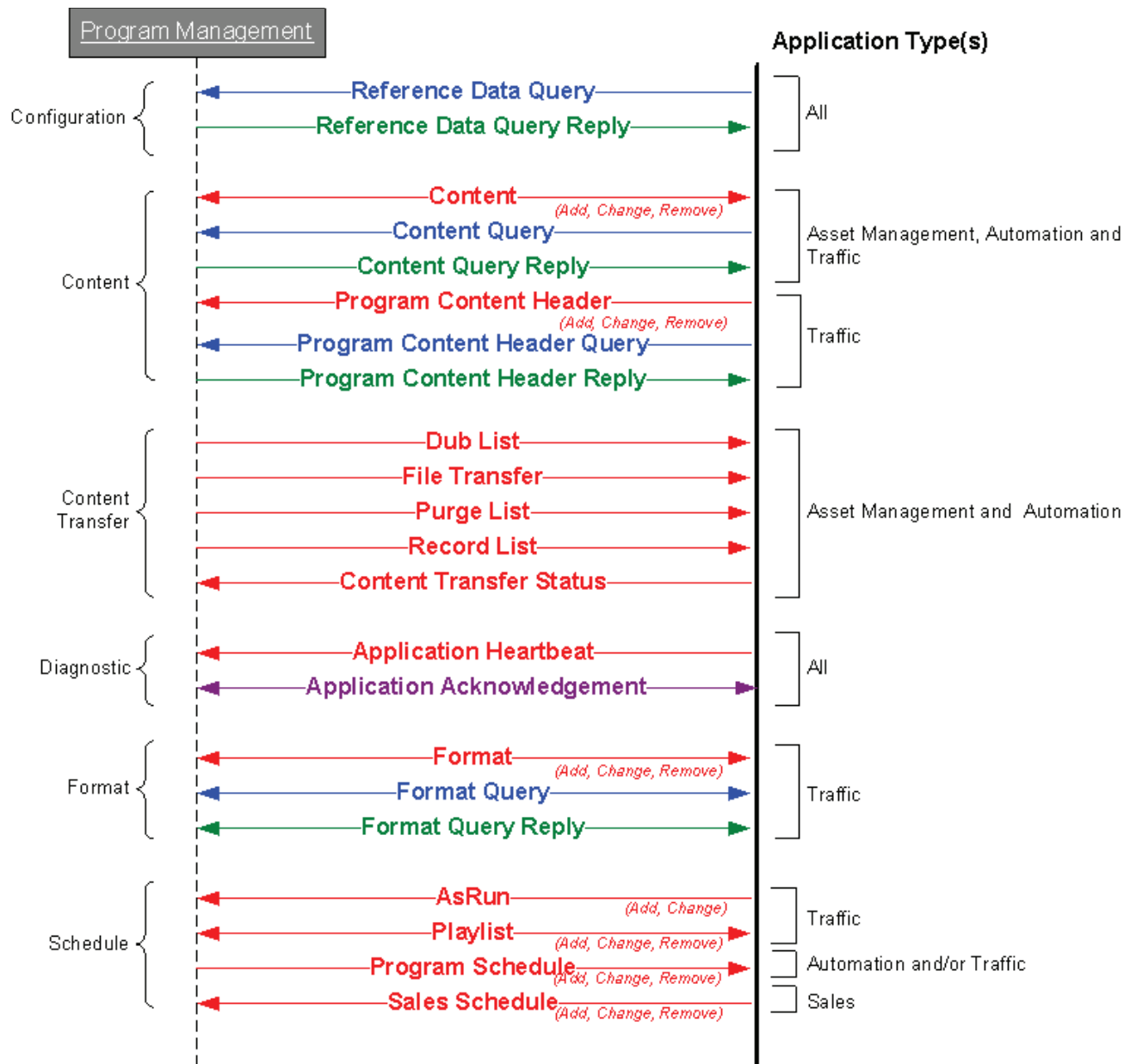
8.2 Automation

The following message usages are recommended for asset automation applications.



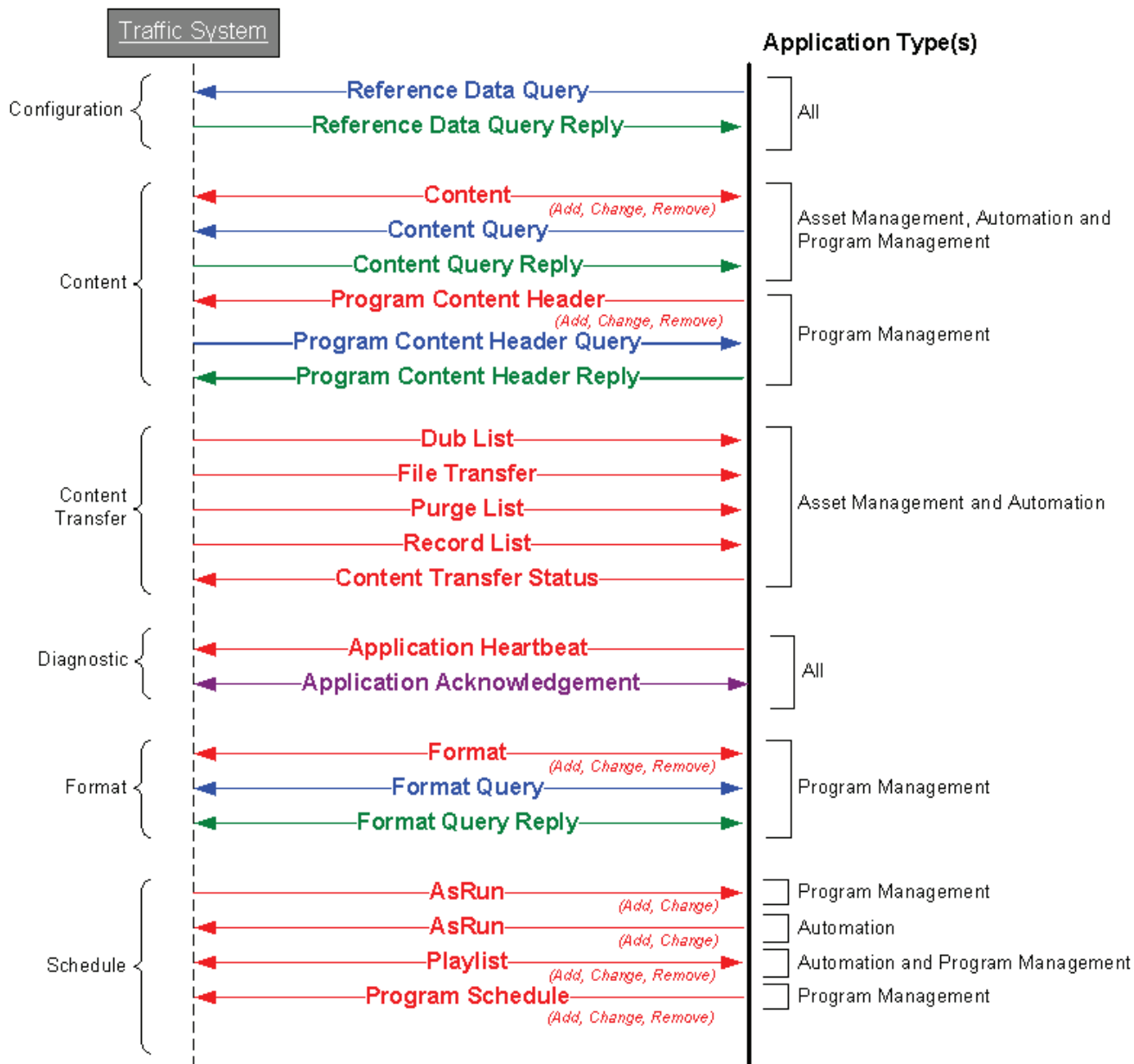
8.3 Program Management

The following message usages are recommended for program management applications.



8.4 Traffic

The following message usages are recommended for traffic applications.



9 Configuration

9.1 Overview (Informative)

The configuration elements within the BXF protocol allow applications to communicate configuration values that are commonly referenced in other types of messages (e.g., Playlist). Channel is a good example, since each application may have a different identifier for a channel, but the channel is in fact the same from the client's perspective. Enumerations can not be used for these values, since they are defined at runtime. Per Section "4 (Glossary/Configuration)" of the BXF documentation:

"There are several schema elements that are not specifically enumerated in the schema and must be configured between the two systems before the elements can be used."

9.1.1 Configuration Processing Conventions (Normative)

It is important to note that the terms "reference data", "configuration", "configuration information" and "configuration values" all represent the same functionality in the following paragraphs and sections.

To optimally support configuration capabilities, one or more applications that participate in the BXF message exchange need to be designated as "sync points" for configuration information. These applications will provide mapping functionality to associate different application values. In addition, they may provide functionality for creating new configuration values and modifying existing values, so that the configuration information can be maintained over time. If the sync point concept is not utilized and instead each application provides its own mapping capabilities, then redundant mapping will be required within each application (this approach is not recommended).

Through the mapping process, sync point applications establish the canonical set of values that will be utilized by all participating applications. These values will be distributed by the sync point application to all participating applications. This means that participating applications will have their own (internal) configuration values and an associated sync point value. Refer to the example below for additional information.

The number of applications acting as sync points will vary based on the applications involved, the configuration types and the client. Each configuration type shall have only one owner (application). This is typically determined by the client, which identifies ownership of the configuration types.

If ownership of a configuration type is managed by an application with sync point functionality then the associated configuration values should be managed from the sync point.

9.1.2 Integrating without a Sync Point (Normative)

If the integrating applications do not have sync point functionality then configuration values may be mapped manually within each application. Configuration values are communicated between the applications in a non-programmatic manner and the applications are updated through existing functionality.

9.1.3 Integrating with a Sync Point (Normative)

9.1.3.1 Participating Application Responsibilities

Each application shall provide the following functionality to support configuration messages:

- Respond to a request for configuration types and values, providing support configuration types and values (if any)
- Process notification messages containing mappings between and application's values and the sync point's values

- Process notification messages containing updated or removed configuration values (optionally supported based on configuration type)
- Respond to a request to add configuration values (optionally supported based on configuration type)

These capabilities allow an application to provide its configuration values and maintain a 1:1 mapping between its values and the canonical values of the sync point. Applications are not required to maintain mappings for every application that they are connected to.

9.1.3.2 Sync Point Application Responsibilities

Sync point applications shall provide the following functionality to support configuration messages:

- Request the configuration values from other applications for one or more configuration types that are owned and process response messages containing each application's configuration values
- Provide a mapping facility (e.g., user interface, algorithms, etc.) to associate application values and identify the canonical set for a given configuration type
- Notify participating applications with the mappings between their values and the canonical values that will be used in BXF messages
- Notify participating applications whenever values are updated or removed
- Request configuration values be added and process response messages containing each application's associated configuration values

These capabilities allow a "sync point" application to identify and maintain the canonical set of values for BXF messaging. The "sync point" applications shall contain their own configuration values as well and these should be mapped just like any other applications values.

9.1.4 Configuration Example (Informative)

The following example shows values for three applications, where applications A and B are participating applications and application C is acting as the sync point for the configuration type of "Channel".

9.1.4.1 Initial Configuration

No mappings exist for the configuration data in applications A, B and C.

Application A			
ID	Value	Mapped ID	Mapped Value
0001	KAAA		
0002	KBBB		
0003	KCCC		

Application B			
ID	Value	Mapped ID	Mapped Value
2001	KA		
2002	KB		
2003	KC		

Application C (Sync Point)				
Canonical ID	Canonical Value	Destination	Application ID	Application Value
		Application C	9010	A
		Application C	9011	B
		Application C	9012	C

9.1.4.2 Request and Process Configuration Values

Application C requests configuration values from applications A and B and processes their responses, adding their configuration values to application C (resulting changes are **highlighted**).

Application A			
ID	Value	Mapped ID	Mapped Value
0001	KAAA		
0002	KBBB		
0003	KCCC		

Application B			
ID	Value	Mapped ID	Mapped Value
2001	KAAA		
2002	KB		
2003	KC		

Application C (Sync Point)				
Canonical ID	Canonical Value	Destination	Application ID	Application Value
		Application C	9010	A
		Application C	9011	B
		Application C	9012	C
		Application A	0001	KAAA
		Application A	0002	KBBB
		Application A	0003	KCCC
		Application B	2001	KAAA
		Application B	2002	KB
		Application B	2003	KC

9.1.4.3 Map Configuration Values

Application C maps the configuration values for the example configuration type "Channel", creating the canonical set of channel values (resulting changes are **highlighted**). Three unique channel values are identified: KAAA, KBBB and KCCC. Note that the mapping activity only occurs in application C. Note also that mapping of the values is one of the responsibilities of the sync point and is accomplished via the "mapping facilities" indicated in Section 9.1.3.

Application A			
ID	Value	Mapped ID	Mapped Value
0001	KAAA		
0002	KBBB		
0003	KCCC		

Application B			
ID	Value	Mapped ID	Mapped Value
2001	KAAA		
2002	KB		
2003	KC		

Application C (Sync Point)				
Canonical ID	Canonical Value	Destination	Application ID	Application Value
1	KAAA	Application C	9010	A
2	KBBB	Application C	9011	B
3	KCCC	Application C	9012	C
1	KAAA	Application A	0001	KAAA
2	KBBB	Application A	0002	KBBB
3	KCCC	Application A	0003	KCCC
1	KAAA	Application B	2001	KAAA
2	KBBB	Application B	2002	KB
3	KCCC	Application B	2003	KC

9.1.4.4 Send Mappings to Participating Applications

Application C sends the mapped values to applications A and B and their mappings are updated accordingly (resulting changes are **highlighted**). Applications A and B provide mapping functionality between their values and the canonical values, but they do not need to support mappings for all values in all applications.

Application A			
ID	Value	Mapped ID	Mapped Value
0001	KAAA	1	KAAA
0002	KBBB	2	KBBB
0003	KCCC	3	KCCC

Application B			
ID	Value	Mapped ID	Mapped Value
2001	KAAA	1	KAAA
2002	KB	2	KBBB
2003	KC	3	KCCC

Application C (Sync Point)				
Canonical ID	Canonical Value	Destination	Application ID	Application Value
1	KAAA	Application C	9010	A
2	KBBB	Application C	9011	B
3	KCCC	Application C	9012	C
1	KAAA	Application A	0001	KAAA
2	KBBB	Application A	0002	KBBB
3	KCCC	Application A	0003	KCCC
1	KAAA	Application B	2001	KAAA
2	KBBB	Application B	2002	KB
3	KCCC	Application B	2003	KC

9.1.4.5 Send Canonical Values on BXF Messages

When applications A, B and C send Format, Playlist, AsRun, Program Schedule, etc. messages, only the canonical values (i.e., KAAA, KBBB and KCCC) are sent for the channel identifier.

- Each application must translate its internal channel identifier into the canonical value when creating messages to send to other applications. This allows the sender to send the same message to all participating applications.
- Each application is responsible for translating the canonical channel identifier into its internal identifier when BXF messages are received and processed.

9.2 Recommended Conventions and Configuration Types (Normative)

Each application shall provide the capability to maintain a 1:1 mapping between its configuration values and the canonical values of the sync point.

The following configuration types are defined within the BXF protocol and are application and message type-specific.

9.2.1 BXF Configuration Types

The following configuration types are specified within the BXF protocol:

Configuration Type	Description
AgencyCode	Agency Code
Applicability	Content, Content Transfer and Schedule messages
Usage	Optional Applications that are agency code aware should provide their configuration values. Schedule-based agency codes are not recommended. Agency code is utilized in: • */Content/NonProgramContent/Details/Agency/AgencyCode • */ContentTransfer/NonProgramContent/Details/Agency/AgencyCode • */Schedule/ScheduledEvent/EventData/PrimaryEvent/NonProgramEvent/Details/Agency/AgencyCode • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/NonProgramEvent/Details/Agency/AgencyCode
AlternateIdType	Type of alternate identifier and the authoritative source of the content (e.g., ISCI, Custom)
Applicability	Content, Content Transfer and Schedule messages
Usage	Recommended Applications that utilize alternate identifiers for content should provide their configuration values. Alternate identifier type is utilized in: • */Content/ProgramContent/ContentMetadata/ContentId/AlternateId/@type • */Content/NonProgramContent/ContentMetadata/ContentId/AlternateId/@idType • */ContentTransfer/Content/ProgramContent/ContentMetadata/ContentId/AlternateId/@idType • */ContentTransfer/Content/NonProgramContent/ContentMetadata/ContentId/AlternateId/@idType • */Schedule/ScheduledEvent/Content/ContentId/AlternateId/@idType
AlternateIdSource	The source of the content that is using the alternative ID type of alternate identifier as the Authoritative Source of the content (e.g., traffic; automation; CDS; program management, ISCI, custom)
Applicability	Content, Content Transfer and Schedule messages
Usage	Recommended Applications that utilize alternate identifiers for content should identify them as the Authoritative Source configuration values. Authoritative source is utilized in: • */Content/ProgramContent/ContentMetadata/ContentId/AlternateId/@authoritativeSource • */Content/NonProgramContent/ContentMetadata/ContentId/AlternateId/@authoritativeSource • */ContentTransfer/Content/ProgramContent/ContentMetadata/ContentId/AlternateId/@authoritativeSource • */ContentTransfer/Content/NonProgramContent/ContentMetadata/ContentId/AlternateId/@authoritativeSource • */Schedule/ScheduledEvent/Content/ContentId/AlternateId/@authoritativeSource

AuthoritativeIdSource	The industry standards body or other source for the identification value listed for the company (e.g., 4A's ID, FCC Call Sign, IRS EIN#, etc.)
Applicability	Job creation, specifically when defining a company and the type of company represented in a specific job such as an agency or advertiser and in describing the Advertiser the produced the nonprogram content or producers that produced the program content.
Usage	Recommended Applications that utilize company identifiers for jobs and content should identify them as the Authoritative Id Source configuration values. Authoritative Id Source is utilized in: • /JobRequester/CompanyDetails/Identifiers/AuthoritativeIdSource • /JobRequester/AdditionalCompany/CompanyDetails/Identifiers/AuthoritativeIdSource • /JobManager/CompanyDetails/Identifiers/AuthoritativeIdSource • /JobPerformers/CompanyDetails/Identifiers/AuthoritativeIdSource • /Content/NonProgramContent/Details/Advertiser/ParentCompany/CompanyDetails/Identifiers/AuthoritativeIdSource • Content/ProgramContent/Series/Producers/CompanyDetails/Identifiers/AuthoritativeIdSource
AssetName	Tape storage format (e.g., BetaSp, DigiBeta, MiniDV)
Applicability	Content, Content Transfer and Schedule messages
Usage	Recommended Applications that are asset name aware should provide their configuration values. Asset name value is utilized in: • */Content/ProgramContent/ContentMetadata/Media/MediaLocation/Location/PhysicalAsset/@assetName • */Content/NonProgramContent/ContentMetadata/Media/MediaLocation/Location/PhysicalAsset/@assetName • */ContentTransfer/Content/ProgramContent/ContentMetadata/Media/MediaLocation/Location/PhysicalAsset/@assetName • */ContentTransfer/Content/NonProgramContent/ContentMetadata/Media/MediaLocation/Location/PhysicalAsset/@assetName • */Schedule/ScheduledEvent/Content/Media/MediaLocation/Location/PhysicalAsset/@assetName
Classification	Program or spot classifications that may restrict the movement of a program or spot from one event to another by the operator
Applicability	Schedule messages
Usage	Optional Applications that are classification aware may provide their configuration values. Classification is utilized in: • */Schedule/ScheduledEvent/EventData/PrimaryEvent/ProgramEvent/Constraints/Rules/Classification • */Schedule/ScheduledEvent/EventData/PrimaryEvent/NonProgramEvent/Constraints/Rules/Classification • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/NonProgramEvents/

	Constraints/Rules/Classification
ContentType	Content Type (e.g., network, local, entertainment, news)
Applicability	Schedule messages
Usage	Optional Applications that are content type aware may provide their configuration values. Content type is utilized in • */Schedule/ScheduledEvent/ContentType • */Schedule/AsRun/CompleteAsRun/ContentType
FederalSource	Federal Source for the event (e.g., live, recorded or network)
Applicability	Schedule messages
Usage	Recommended Applications that are federal source aware should provide their configuration values. Federal source is utilized in: • */Schedule/ScheduledEvent/EventData/FederalSource
FederalType	Some countries require further codification of the content such as the FCC Type or CRTC Code. This is typically a specialized value limited by the countries regulatory body. (e.g., CM, PR, PSA, PRO, PRC; etc.)
Applicability	Schedule messages
Usage	Recommended Applications that are federal type aware should provide their configuration values. Federal type is utilized in: • */Schedule/ScheduledEvent/EventData/FederalType
JobInstructionType	Instruction type associated with a specific job and which is specific to the type of jobs and systems being used
Applicability	Job messages
Usage	Recommended Applications that use job details with specific instructions for multiple jobs should provide their configuration values. Job Instruction Type is utilized as an attribute in: • */JobDetail/JobInstruction/instructionType
MacroName	Macro Name
Applicability	Content, Content Transfer, Format and Schedule messages
Usage	Optional Applications that are macro aware may provide their configuration values. Macro name is utilized in: • */Content/ProgramContent/ContentMetaData/Media/ProfileMacro/MacroName • */Content/NonProgramContent/ContentMetadata/Media/ProfileMacro/MacroName • */ContentTransfer/Content/ProgramContent/ContentMetadata/Media/ProfileMacro/MacroName • */ContentTransfer/Content/NonProgramContent/ContentMetadata/Media/Profile

	eMacro/MacroName • */Format/FormatStructure/FormatElements/NonPrimaryElements/Macros • */Schedule/ScheduledEvent/EventData/MacroEvent/MacroName • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/Macros/MacroName
MessageDestination	Intended destination of a BXF message
Applicability	All messages
Usage	Optional The destination values are configured during system setup. Destination values are not mapped so they are not a recommended Configuration type. Destination is utilized in: • BxfMessage/@destination
MessageOrigin	The origin of a BXF message
Applicability	All messages
Usage	Optional The origin values are configured during system setup. Origin values are not mapped so they are not a recommended configuration type. Origin is utilized in: • BxfMessage/@origin
NonPrimaryEventName	Non-primary event type (e.g., key, GPI, tone, logo, bug, crawl)
Applicability	Schedule messages
Usage	Recommended Applications that are non-primary event name aware should provide their configuration values. Non-primary event name is utilized in: • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/NonPrimaryEventName
OverlayOpportunityType	Used to define the types of overlays that can be used such as crawls, voice-overs, over the shoulders, logos, etc, and is system dependent
Applicability	Schedule messages and content messages
Usage	Recommended Applications that describing specific overlays in detailed traffic, automation and scheduling situations. OverlayOpportunityType is an attribute used in content and schedule areas: • */Schedule/ScheduledEvent/Content/ContentDetail/ProgramContent/Elements/Element/ProgramElement/OverlayOpportunity/opportunityType • */Schedule/ScheduledEvent/Content/ContentDetail/ProgramContent/Elements/Element/OverlayOpportunity/opportunityType • */Content/ProgramContent/Elements/Element/ProgramElement/OverlayOpportunity/opportunityType • */Content/ProgramContent/Elements/Element/OverlayOpportunity/opportunityType
ProductCode	Product Code that categorizes a product (e.g., fast food, auto, beverages)
Applicability	Content, Content Transfer and Schedule messages

Usage	Optional Applications that are product code aware may provide their configuration values. Product code is utilized in: • */Content/NonProgramContent/Details/Product/ProductCode • */ContentTransfer/Content/NonProgramContent/Details/Product/ProductCode • */Schedule/ScheduledEvent/EventData/PrimaryEvent/NonProgramEvent/Details/Product/ProductCode • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/NonProgramEvent/Details/Product/ProductCode
ProtectionSourceName	Name of the type of protection source to be used (e.g., ProtSrc1)
Applicability	Schedule messages
Usage	Recommended Applications that are protection source aware should provide their configuration values. Protection source is utilized in: • */Schedule/ScheduledEvent/EventData/Protection/ProtectionSourceName
RouterSourceName	Router Source Name (e.g., Router1, Srv2)
Applicability	Content, Content Transfer and Schedule messages
Usage	Recommended Applications that are router source aware should provide their configuration values. Router source is utilized in: • */Content/ProgramContent/ContentMetadata/Media/MediaLocation/Location/RouterSource/Name • */Content/NonProgramContent/ContentMetadata/Media/MediaLocation/Location/RouterSource/Name • */ContentTransfer/Content/ProgramContent/ContentMetadata/Media/MediaLocation/Location/RouterSource/Name • */ContentTransfer/Content/NonProgramContent/ContentMetadata/Media/MediaLocation/Location/RouterSource/Name • */Schedule/ScheduledEvent/Content/Media/MediaLocation/Location/RouterSource/Name
SpotType (NonProgramDetailType)	Spot Type for non-program content (e.g., barter spot, promo, PSA, ID, comments)
Applicability	Content, Content Transfer and Schedule messages
Usage	Recommended Applications that are spot type aware should provide their configuration values. Spot type is utilized in: • */Content/NonProgramContent/Details/SpotType • */ContentTransfer/Content/NonProgramContent/Details/SpotType • */Schedule/ScheduledEvent/EventData/PrimaryEvent/NonProgramEvent/Details/SpotType • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/NonProgramEvents/Details/SpotType

SpotSalesClassification	Spot rate section for non-program content (e.g., P1, R2, NP, comments)
Applicability	Schedule messages
Usage	Optional Applications that are spot sales classification (i.e., “rate section”) aware may provide their configuration values. Spot sales classification is utilized in: • */Schedule/ScheduledEvent/EventData/PrimaryEvent/NonProgramEvent/SpotSalesClassification • */Schedule/ScheduledEvent/EventData/NonPrimaryEvent/NonProgramEvents/SpotSalesClassification
TransitionPattern	Available patterns for transitions from one video source to another (Fade, Wipe)
Applicability	Schedule messages
Usage	Recommended Applications that are video transition pattern aware should provide their configuration values. Transition pattern is utilized in: • */Schedule/ScheduledEvent/EventData/Transitions/VideoTransitions/TransitionPattern

9.2.2 Recommended Extension Types

The following configuration types are recommended for the BXF protocol:

Condition	Result
AspectRatio	Identifies the aspect ratio for the video (e.g., 16:9, 4:3)
Applicability	Content, Content Transfers, Schedules
Usage	Recommended Applications that are Aspect Ratio aware should provide their Aspect Ratio configuration values. BXF does not provide a configurable element for aspect ratio, so these values should be provided in the following location: • */Content/ProgramContent/ContentMetadata/Media/PrivateInformation/AspectRatio • */Content/NonProgramContent/ContentMetadata/Media/PrivateInformation/AspectRatio • */ContentTransfer/Content/ProgramContent/ContentMetadata/Media/PrivateInformation/AspectRatio • */ContentTransfer/Content/NonProgramContent/ContentMetadata/Media/PrivateInformation/AspectRatio • */Schedule/ScheduledEvent/Content/Media/PrivateInformation/AspectRatio
BreakCode	Codes to allow systems to configure breaks for certain types of spots (e.g. Promo Only, Barter, etc.)
Applicability	Format
Usage	Recommended

	Applications that are BreakCode aware should provide their Break Code configuration values. BreakCode value is utilized within: • BxfMessage/BxfData/Format/FormatStructure/FormatElements/PrivateInformation/BreakCode
Channel	A unique identifier for a channel
Applicability	Content, Content Transfers, Schedules
Usage	Recommended Applications that are Channel aware should provide their Channel configuration values. Channel value is utilized within: • BxfMessage/BxfData/Schedule/Channel/Name element on schedule messages • BxfMessage/BxfData/Content/ProgramContent/ContentMetaData/Policy/AssignedChannels on program content messages • BxfMessage/BxfData/Content/NonProgramContent/ContentMetaData/Policy/AssignedChannels on non-program content messages
Distributor	Distributor of the content; related to the contract data (e.g., Sony, Warner)
Applicability	Content, Program Contracts
Usage	Recommended Applications that are Distributor aware for content should provide their Distributor configuration values. Distributor value is utilized within: • */Content/ProgramContent/Contract/Distributor • */Content/ProgramContract/Distributor
Genre	Type (or grouping) of content based on the storyline or style of the content. (e.g., police drama, western, game show, children's show)
Applicability	Content, Content Transfers, Schedules
Usage	Recommended Applications that are Content aware should provide their Genre configuration values. Genre value is utilized within: • */ContentMetaData/Genre
ParentalRatingRegion	Parental Rating Region for the content (e.g., 01 – United States, 02 – Canada (*PMCP Values are used))
Applicability	Content, Formats (Private Information)
Usage	Recommended Applications that are Parental Rating Region aware for content should provide their Parental Rating Region configuration values. Parental Rating Region value is utilized within: • */Content/ProgramContent/ParentalRating/@region • */Format/PrivateInformation/@region
ParentalRatingDimension	Parental Rating Dimension for the content (e.g., TV Parental)
Applicability	Content, Format (Private Information)

Usage	Recommended Applications that are Parental Rating Dimension aware for content should provide their Parental Rating Dimension configuration values. Parental Rating Dimension value is utilized within: • */Content/ProgramContent/ParentalRating/@dimension • */Format/PrivateInformation/@dimension
ParentalRatingValue	Parental Rating Value for the content (e.g., TV-G, TV-PG)
Applicability	Content, Format (Private Information)
Usage	Recommended Applications that are Parental Rating Value aware for content should provide their Parental Rating Value configuration values. Parental Rating Region value is utilized within: • */Content/ProgramContent/ParentalRating/@value • */Format/PrivateInformation/@value
QualityRatingRegion (Quality = number of stars can use .5 for half stars.)	Quality Rating Region for the content (e.g., 01 – United States, 02 – Canada (*PMCP Values are used))
Applicability	Content
Usage	Recommended Applications that are Quality Rating Region aware for content should provide their Quality Rating Region configuration values. Parental Rating Region value is utilized within: • */Content/ProgramContent/QualityRating/@RatingRegion
QualityRatingDimension	Quality Rating Dimension for the content (e.g., Chicago Tribune)
Applicability	Content
Usage	Recommended Applications that are Quality Rating Dimension aware for content should provide their Quality Rating Dimension configuration values. Quality Rating Dimension value is utilized within: • */Content/ProgramContent/QualityRating/@Dimension
RightsType	The type of program rights; related to the contract data (e.g., view-based, traditional broadcast, box office)
Applicability	Program Contracts
Usage	Recommended Applications that are Rights Type aware for content should provide their Rights Type configuration values. . Rights Type value is utilized within: • */Content/ProgramContract/Rights/RightsType

9.3 Requesting Configuration Information (Queries) (Normative)

The following sections identify how an application can request the configuration of another application. Each section is annotated, indicating if the functionality is required, optional or to be avoided by applications.