International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.1022
(07/2016)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

Multimedia Quality of Service and performance – Generic and user-related aspects

---

Buffer models for media streams on TCP transport

Recommendation ITU-T G.1022

ITU-T

International Telecommunication Union

ITU-T G-SERIES RECOMMENDATIONS

TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100–G.199
GENERAL CHARACTERISTICS COMMON TO ALL ANALOGUE CARRIER-TRANSMISSION SYSTEMS	G.200–G.299
INDIVIDUAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON METALLIC LINES	G.300–G.399
GENERAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON RADIO-RELAY OR SATELLITE LINKS AND INTERCONNECTION WITH METALLIC LINES	G.400–G.449
COORDINATION OF RADIOTELEPHONY AND LINE TELEPHONY	G.450–G.499
TRANSMISSION MEDIA AND OPTICAL SYSTEMS CHARACTERISTICS	G.600–G.699
DIGITAL TERMINAL EQUIPMENTS	G.700–G.799
DIGITAL NETWORKS	G.800–G.899
DIGITAL SECTIONS AND DIGITAL LINE SYSTEM	G.900–G.999
MULTIMEDIA QUALITY OF SERVICE AND PERFORMANCE – GENERIC AND USER-RELATED ASPECTS	**G.1000–G.1999**
TRANSMISSION MEDIA CHARACTERISTICS	G.6000–G.6999
DATA OVER TRANSPORT – GENERIC ASPECTS	G.7000–G.7999
PACKET OVER TRANSPORT ASPECTS	G.8000–G.8999
ACCESS NETWORKS	G.9000–G.9999

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

Recommendation ITU-T G.1022

Buffer models for media streams on TCP transport

Summary

Recommendation ITU-T G.1022 defines buffer models that predict the behaviour of client side buffers for audio/video streams. Buffer models are client independent and receive A/V data, meta information and network state information as input, as output encoded frames and playback event information are given. The buffer models are intended to be used for the development of client performance metrics.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T G.1022	2016-07-29	12	11.1002/1000/12967

Keywords

Buffer model, HTTP, progressive download video, pseudo streaming, TCP streaming, video streaming.

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

FOREWORD

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

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

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

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

NOTE

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

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

INTELLECTUAL PROPERTY RIGHTS

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

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

Table of Contents

Introduction

This Recommendation is organized as follows: Beyond the Scope and usual introductory material, clauses 6 to 12 describe the modelling approach and different aspects of the buffer models. It is possible to use the model with minimal measured and supplied information, and clause 7 describes this in detail. Annexes outline specific media flow and connection conditions, such as single media flow on a single reliable connection. Each annex gives the specific requirements for model operation and defines default values for input parameters.

Recommendation ITU-T G.1022

Buffer models for media streams on TCP transport

1 Scope

This Recommendation defines models to estimate the buffer occupancy and operational state used in client media players employing TCP transport and other forms of reliable transport.

The measurement point(s) provide an observation point on TCP connection(s) primarily within the end-to-end path, where input data are collected.

The scope of work can be described in terms of inputs and outputs of the model(s):

Exact procedures for processing network layers and subsequent container processing are beyond the scope of this Recommendation, as it is highly variable and obviously must be matched to the specific circumstances. However, general guidance for readers is provided.

Inputs – Frames or partial frames (audio/visual (A/V) frames) pertaining to the reliable transport connection(s) in use and the time stamp when each packet appears at the measurement point. Normal operation considers both directions of transmission. Alternatively, meta-data from the container or streaming media application layer and network layers may be used whenever present.

Outputs – for each model state change, it is required to report the new state name, the timestamp of the change in units of measure equal to 1 millisecond resolution. Optionally, the current buffer fill level in units of milliseconds is reported for each change. There is optional periodic reporting of the buffer fill level with timestamp and current state name. Also, the frame type and size information that is played-out may be reported and for some models, the complete media stream information. Furthermore, if there are multiple video streams (e.g., in the adaptive case) a possible stream change could be reported. When available for each frame, there shall be a descriptive tuple containing a synthesized or decoded status, frame type, frame size in bytes, duration in time (ms) represented by the frame, and the payload bytes. The frame type should be registered (somewhere) and referenced in the type value.

Operation – there are scenarios where it may not be possible to access the media frame boundaries within the A/V stream, such as when the media is encrypted or an un-decodeable media format is encountered. In such cases, the set of default frame tuples within a container may be synthesised, and the distinction between decoded frame boundaries and synthesised frames shall be indicated. When a sufficiently long pause is detected in the TCP stream (or media flow) and a waiting time for resumption of the TCP stream expires, the monitoring system declares an End of stream if the buffer is coincidentally depleted, and normal operation should cease.

All stored state, including buffer state and queuing diagnostic information may be made available for on demand retrieval. Some network analysis is allowed to assist the mid-path monitoring to determine frame arrival time and other signals that are inputs to the model (the details of complex network interactions are out of the scope of the Recommendation, but some general guidance is provided).

Advertisements interjected in user sessions are extremely important to the business of video delivery, and evaluation of their delivery may need to be separate to meet the needs of performance management.

NOTE – Non-reliable and UDP streams are outside of the scope of this Recommendation; they are covered in [b-ITU-T G.1021].

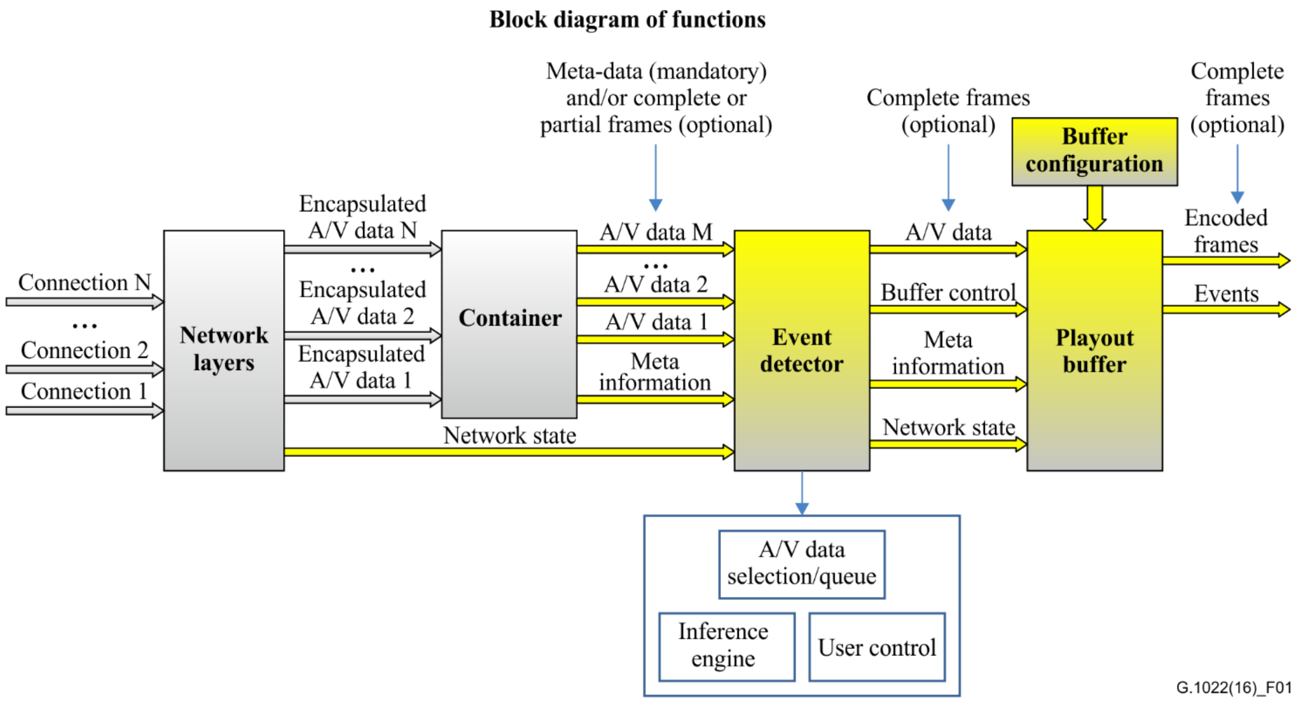

Figure 1-1 – Block diagram of functions

Adaptive streaming is in scope, and as the table below shows, all forms known at this time use TCP and are possible topics of coverage for this Recommendation.

Table 1-1 – Forms of streaming using TCP

Type	Source	TCP
Adaptive http Streaming (AHS)	Defined in 3GPP Release 9	X
http Adaptive Streaming (HAS)	Defined in Open IPTV Forum Release 2	X
Dynamic Adaptive Streaming over HTTP (DASH)	Developed by MPEG, and developing beyond AHS and HAS. Adopted by 3GPP R10.	X
http Dynamic Streaming	Developed by Adobe Systems	X
http Live Streaming (HLS)	Developed by Apple	X
Microsoft Smooth Streaming	Developed by Microsoft	X

2 References

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

[ITU-T G.9960] Recommendation ITU-T G.9960 (2011), *Unified high-speed wireline-based home networking transceivers – System architecture and physical layer specification*.

[ITU-T H.262] Recommendation ITU-T H.262 (2012) | ISO/IEC 13818-2:2013, *Information technology – Generic coding of moving pictures and associated audio information: Video*.

[ITU-T I.113] Recommendation ITU-T I.113 (1997), *Vocabulary of terms for broadband aspects of ISDN*.

[ITU-T J.123] Recommendation ITU-T J.123 (2002), *Multiplexing format for webcasting on the TCP/IP network*.

[ITU-T J.124] Recommendation ITU-T J.124 (2004), *Multiplexing format for multimedia webcasting over TCP/IP networks*.

[ITU-T P.1202] Recommendation ITU-T P.1202 (2012), *Parametric non-intrusive bitstream assessment of video media streaming quality*.

[ITU-T Y.1540] Recommendation ITU-T Y.1540 (2016), *Internet protocol data communication service IP packet transfer and availability performance parameters*.

[ITU-T Y.2770] Recommendation ITU-T Y.2770 (2012), *Requirements for deep packet inspection in next generation networks*.

[IETF RFC 793] IETF RFC 793 (1981), *Transmission Control Protocol*.

[IETF RFC 2460] IETF RFC 2460 (1998), *Internet Protocol, Version 6* (IPv6).

[IETF RFC 2616] IETF RFC 2616 (1999), *Hypertext Transfer Protocol Version — HTTP/1.1*.

[IETF RFC 2818] IETF RFC 2818 (HTTP Over TLS, 2000), *Hypertext Transfer Protocol Version Over TLS*.

[IETF RFC 3550] IETF RFC 3550 (2003), *RTP: A Transport Protocol for Real-Time Applications*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 rebuffering artefacts [ITU-T P.1202]

3.1.2 payload [ITU-T Y.2770]

3.1.3 flow [ITU-T G.9960]

3.1.4 constant bitrate coded video [ITU-T H.262]

3.1.5 variable bitrate [ITU-T H.262]

3.1.6 chunk [ITU-T J.123], [ITU-T J.124]

3.1.7 block [ITU-T I.113]: A unit of information consisting of a header and an information field.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

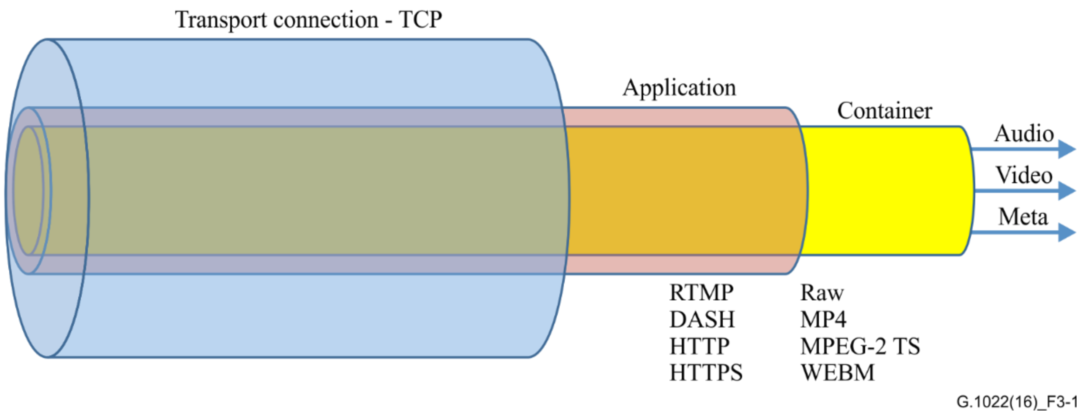

Figure 3-1 – Terms defined in this Recommendation

3.2.1 audio and video frames: Mandatory: arrival time at client or measurement point, frame duration or frame rate (from meta-data).

Optional: type, frame size, PTS, and/or DTS

3.2.2 overall meta-data: The meta-data that can be found in various places, such as the container meta-data and the manifest files.

Mandatory: frame rate, or mean frame rate calculated from session duration for encrypted streams

Optional: The total number of media frames, encoder information, encoder profile and resolution, number of audio channels, encoder bit stream rate

3.2.3 manifest files: Files delivered separately from the media container and elementary stream that optionally describe operating points and provide a list where chunks or segments of a media file can be downloaded or streamed.

3.2.4 container layer: Optionally (raw data may be provided), the layer which encapsulated the elementary streams. Examples include, MP4, MPEG2-TS, WEBM.

3.2.5 streaming media application layer: Aggregates data from network connections and provides raw data or data in a container. May optionally include manifest files.

Examples include HTTP, HTTPS, DASH, HLS.

3.2.6 TCP layer connection: See [IETF RFC 793] TCP endpoints initialize and maintain certain status information for each data stream. The combination of this information, including sockets, sequence numbers, and window sizes, is called a connection. Each connection is uniquely specified by a pair of sockets identifying its two sides.

3.2.7 session: The time from a client selection of a media to completion or premature disconnect, including user control, inferred or explicit.

3.2.8 DASH: Dynamic adaptive streaming over HTTP is the first adaptive streaming solution which has become an International Standard.

3.2.9 frame rate: The number of (progressive) frames displayed per second (fps).

3.2.10 video chunk: A contiguous set of samples for one track of a video.

3.2.11 video block: A unit of a video consisting of a header and an information field.

3.2.12 play-out: The process of transferring buffered information to a player, such as the frame information of a video stream.

3.2.13 play-out buffer: Transient memory where the video is stored for play-out.

3.2.14 stall: A condition occurring at the presentation layer, when media play-out is suspended. This condition typically occurs due to play-out buffer depletion.

3.2.15 re-buffering: A condition occurring in the media buffer when the fill level is sufficiently depleted and buffer exhaust is imminent. This condition typically results in a suspended video play-out.

3.2.16 initial buffering: A condition occurring in the media buffer on the initiation of a media stream, and completing when the configured buffer fill level is accumulated. This condition typically occurs before the start of media play-out.

3.2.17 video buffering and play-out: A condition beginning when stream is received and subsequently the video presentation commences.

3.2.18 single-connection: Only one data stream is used to transmit the media to the client.

3.2.19 multi-connection: Multiple data flows are used in parallel to transmit the media to the client.

3.2.20 buffer control: The state transition described in the table below are mostly user initiated. The inference engine, which is a part of the event detector, can also detect and initiate state changes.

Table 3-1 – State transitions

Old state	New state	Transition signal
Initial uffering	Playing	User pressed play signal
Initial buffering	Paused	User paused signal, or pause detected signal
Initial buffering	Stopped	User terminated signal, or detected end of session signal
Playing	Initial buffering	User sought signal, or seek detected signal
Playing	Paused	User paused signal, or pause detected signal
Playing	Stopped	User terminated signal, or detected end of session signal
Paused	Initial buffering	User sought signal, or seek detected signal
Paused	Playing	User pressed play signal, or play detected signal
Paused	Stopped	User terminated signal, or detected end of session signal
Re-buffering	Initial buffering	User sought signal, or seek detected signal
Re-buffering	Playing	User pressed play signal
Re-buffering	Stopped	User terminated signal, or detected end of session signal

3.2.21 buffer configuration parameters: Primarily, the buffer parameter thresholds (Initial Buffering, Re-buffering, Empty Buffer, Buffer Overflow).

Parameter sets may change over time:

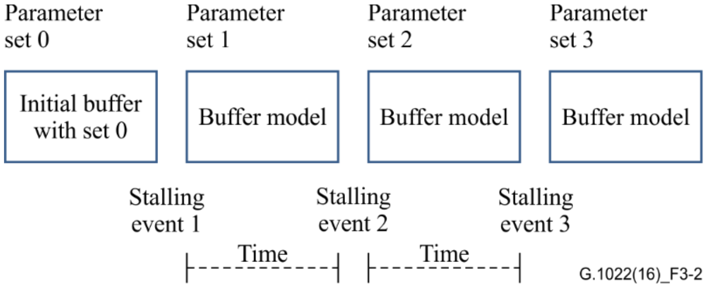

Figure 3-2 – Parameter sets changed over time

A reference set of parameters shall be provided, to encourage equivalence of results.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

AHS	Adaptive HTTP Streaming
API	Application Programming Interface
A/V	Audio/video
BMM	Buffer Modelling Modes
DASH	Dynamic Adaptive Streaming over HTTP
DTS	Digital Theater Systems
FPS	Frames displayed per Second
HLS	HTTP Live Streaming
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
HW	Hardware
MPD	Media Presentation Description
PTS	Presentation Timestamp
RTP	Real-time Transport Protocol
RTT	Retransmission Time
TCP	Transmission Control Protocol
XML	Extensible Markup Language

5 Conventions

None.

6 Description of possibilities for TCP connection

This clause defines the taxonomy of connection types and media flow construction options.

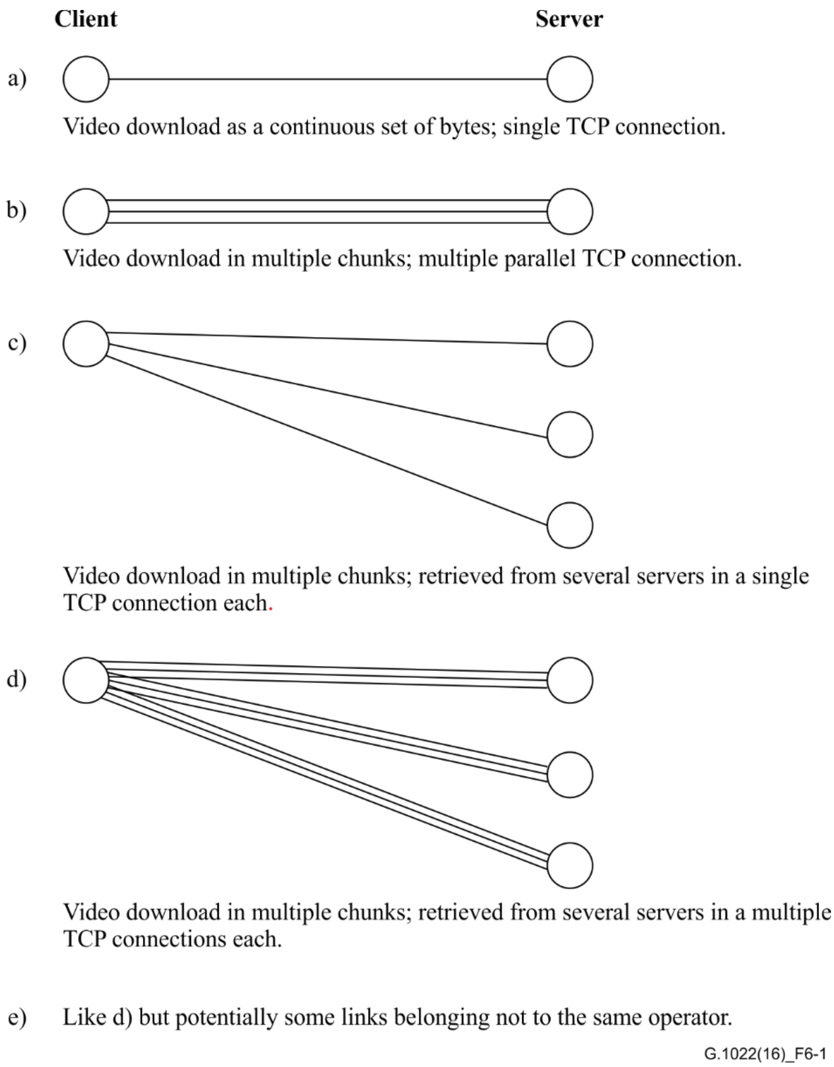

Figure 6-1 – Variants of TCP-based video data downloads (each connection line represents a TCP session)

- Assumption: a) to d) are performed within a single operator; therefore, access to all TCP sessions is required
- However, clients could possibly be multi-homed such that single operator monitoring no longer sees all TCP sessions, therefore monitoring of the play-out buffer within the client device can be a solution.
- In case of adaptive video streaming, especially in DASH, a buffer monitoring mechanism needs to know the structure of the monitored video in order to correctly interpret the observed HTTP downloads of the video segments. Segments can potentially be provided by different servers or even be embedded as a subsection within a larger video file.

7 Container layer processing and frame count estimation

Container layer processing can be performed in several modes achieving different grades of accuracy depending on the available input information as well as available processing power. Clause 7.1 describes the respective buffer modelling modes (BMM).

7.1 Buffer modelling modes

The purpose of a buffer model is the calculation of buffer fill levels (e.g., expressed in buffered play-out seconds or buffered video frames), buffer events and optionally the extraction of raw video frames from either the A/V data and/or network connection information (arrival time stamps) and/or container information (video time stamps) provided.

Buffer modelling can be performed either in a centralized fashion, where all functions are located close to each other, or in a distributed fashion where some functions are placed across the network whereas others remain centralized (datacentre processing or at the client side). In that way, the standard also considers the amount of data exchanged between functional blocks.

The two buffer modelling modes are described in detail below (Figures 7-1 and 7-2).

7.1.1 BMM0 buffer model without frame information

BMM0 is the most primitive mode of buffer level estimation which operates with only streaming media application layer information in order to estimate the buffer fill level and buffer events with coarse-grained precision.

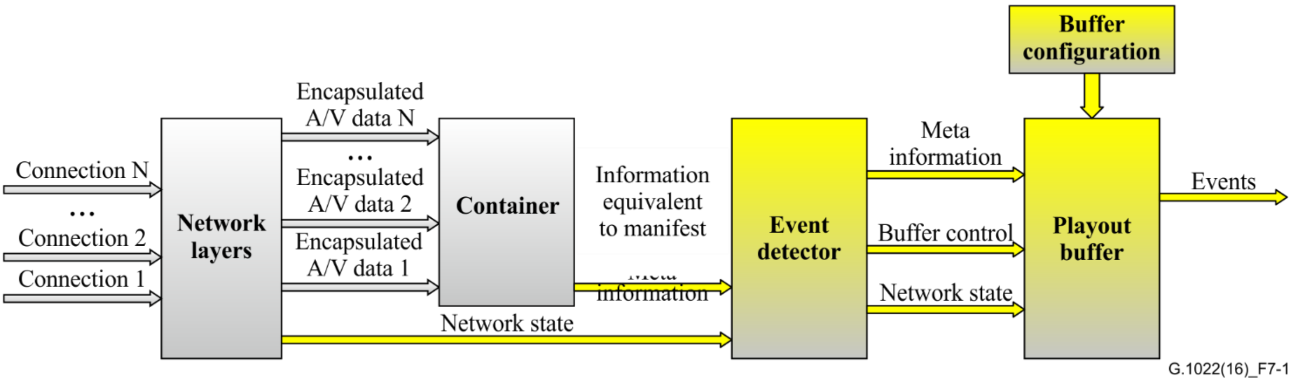

Figure 7-1 – Buffer model without frame information

Input:

– Network connection information
  - Media frame arrival time stamps

– Streaming media application layer information
  - Media chunk duration
  - Starting PTS of the media chunk (can be inferred from manifest files if durations are present for prior media chunks)

Output:

Buffer events, buffer fill level expressed in buffered play-out seconds

Example of use case:

– The streaming media application protocol manifest (e.g., HLS) is available in an unencrypted form but the media container layer is encrypted.

7.1.2 BMM1 buffer model with full container meta-information and media frame byte offsets

The next BMM has access the media container meta-information and byte offsets of individual media frames.

Information about the A/V data, but not necessarily the fully decoded frames, is available.

Elementary stream information may be used in cases where the container meta-information is incomplete or inaccurate (e.g., frame rate changes are signalled in the elementary stream, or PTS for individual frames must be reconstructed from the elementary stream).

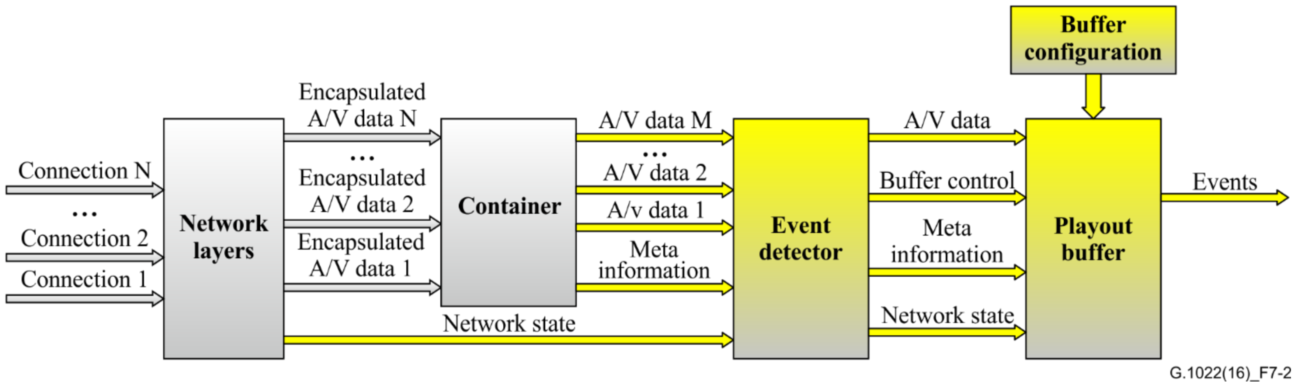

Figure 7-2 – Buffer model with full container meta-information and media frame byte offsets

Input:

– Network connection information
  - Media frame arrival time stamps

– Container information
  - Per-frame PTS (DTS may be used but accuracy may suffer)

– Per-frame size in bytes (Optional) Elementary stream information
  - Frame type
  - Frame rate

Output:

– Buffer events, buffer fill level in seconds/frames

– Video frames (encoded or virtual)

– (Optional) elementary stream configuration parameters

Example of use case:

– The media container layer is unencrypted and media frame byte-offset information is available.

Switching between these BMMs can be triggered by available input parameters. This requires that either frames or container information be available.

Depending on the BMM, the output event data will inform the recipient whether any frame information is available and whether the frame information is encoded or virtual. Furthermore, the BMM will inform the recipient of the buffer fill level time unit.

8 Inference engine

8.1 Overview of inference engine

Inference engine infers (detects) events, such as seek, pause, play, stream switch, and end of session based on A/V data, meta-information and network state. In a multiple connection use case, the pause, play and end inferences need to include all connections. With multiple connections, it is also possible to infer user seeks, and it is necessary to estimate which A/V data source is the current A/V data source. The current A/V data source is special since it is this source that forwards A/V data to the play-out buffer. Moreover, when the inference engine decides to switch the current A/V data source, the play-out buffer needs to be flushed.

For the inference engine, the following parameters can be configured:

Table 8-1 – Configuration variables

Variable	Range	Typical values
DATA_SOURCE_INACTIVITY_TIMEOUT_SEC	1-60	5-30
MIN_SEEK_ACCUMULATED_DURATION_SEC	1-60	5-30
MIN_FORWARD_SEEK_PTS_DELTA_SEC	1-60	5-30
MIN_BACKWARD_SEEK_PTS_DELTA_SEC	1-60	2-30
PAUSE_DETECTION_INACTIVE_DURATION_SEC	1-60	10-20
MIN_SOURCE_SELECTION_ACCUMULATED_DURATION_SEC	1-60	5-10
MIN_GAP_SKIP_WAIT_TIME_SEC	1-60	2-10

8.2 Seeking events during media play-out

The play-out of media streams can be interactively changed upon user request when seeking a new relative play-out time position of the stream.

There are several cases to be distinguished. Firstly, the seeking event can be either forward or backwards with respect to the currently playing presentation time. Secondly, the seeking event can be either within or beyond the time span currently buffered in the play-out buffer.

From a quality monitoring perspective, such seek events can mislead the play-out buffer fill level estimation quite severely. In the forward-seeking case, the fill level estimate is too optimistic and in the backward-seeking case, the estimate is pessimistic. In both situations, the accuracy of the estimation result is detrimentally affected.

This influence is even more severe when the seeking happens to occur within the buffered time span since most players do not send any additional media requests or other indications towards the server, which could in turn be observed and interpreted by the monitoring system. Thus, seeking events within the time span currently buffered in the play-out buffer (either in a forwards or backwards direction relative to the currently-playing presentation time) cannot be detected.

The cases of forward- or backward-seeking events beyond the time span currently buffered in the play-out buffer are obvious to the monitoring system due to required media retrieval requests being issued by the player. The observation of such events serves as a clear indication of such seek events and should in turn cause a reset of the monitoring state to the initial buffering state. Hence, these cases are inherently addressed by this Recommendation and are thus covered by it. The cases in which the seeking event occurs within the time span currently buffered in the play-out buffer, however, are outside of the scope for this Recommendation.

It is also important to distinguish between actual seeking events and fetches of contiguous (or near-contiguous) media chunks. It is important, therefore, to avoid modelling a seeking event whenever the start PTS of a new media chunk is contiguous with the current play-out buffer or is not significantly in the past with respect to the play-out buffer's last buffered frame PTS.

8.2.1 Inputs to the inference engine

– A/V meta-information, described elsewhere;

– Evaluation point, measured in "wall-clock" time;

– Per-frame arrival time, measured in "wall-clock" time;

– State of client-server A/V connections (at least one active, all inactive, none);

– Number of audio samples (a number or unknown). Can be obtained from container or application layers;

– Number of video samples (a number or unknown). Can be obtained from container or application layers.

8.2.2 Outputs to the buffer

– Pause Detected at time T: This occurs when no new A/V data arrive for more than PAUSE_DETECTION_INACTIVE_DURATION_SEC seconds in all connections, the all A/V data connections are inactive, and both the client and the server continue sending and receiving other data (both have at least one active connection).

– Resume Detected at time T: This occurs when in the Paused state and at least one of the connections contributing to the A/V data starts receiving more data.

– Detected end of session at time T: This can occur if all of the connections contributing to the A/V data are closed (terminated) or A/V data reached last expected sample.

– Seek Detected at time T: This occurs when the current source of A/V data is terminated (at the application layer or below) or stops receiving new data for more than DATA_SOURCE_INACTIVITY_TIMEOUT_SEC seconds and a new A/V data source begins or has already begun and continues to be active, and has already received MIN_SEEK_ACCUMULATED_DURATION_SEC seconds. In detecting forward seeks, the absolute PTS difference between the last frame in current A/V data and the first frame in the new A/V data has to be greater than MIN_FORWARD_SEEK_PTS_DELTA_SEC seconds. In detecting backward seeks, the absolute PTS difference between the first unplayed frame in current A/V data and the first frame in the new A/V data has to be greater than MIN_BACKWARD_SEEK_PTS_DELTA_SEC seconds.

In addition to detecting seeks, the inference engine changes its state in ways which are not passed to the play out buffer. These changes are summarized in the following points:

– Initially, there is no current A/V data source selected, but it is possible that there are multiple A/V data sources started by the player in the beginning. The first current A/V data source is chosen from multiple sources by selecting the earliest starting source that has buffered more than MIN_SOURCE_SELECTION_ACCUMULATED_DURATION_SEC seconds of media duration.

– Once the current A/V data source goes inactive, a new one is selected based on the smallest PTS difference between the last frame in current source and any of the frames in the alternate A/V data source, provided it does not meet the seek criteria.

– Whenever the current A/V data and new sources contain A/V data overlapping with respect to PTS, provided it does not meet seek criteria, overlapping data is disregarded.

– Whenever the current A/V data and the alternate A/V data sources do not have continuous PTS and the alternates are not estimated to be seeks (i.e., there is a small gap), the inference engine needs to wait MIN_GAP_SKIP_WAIT_TIME_SEC seconds before skipping over the gap and choosing the new source provided it is active.

– Whenever multiple A/V data sources are receiving data at the same time, only the data from the current A/V data source are moved into the play-out buffer. The data for the other A/V data sources are queued in the A/V data queue.

9 Event detector/data selection/queue

If the event detector component of the buffer model is operating alone, it should de-multiplex the audio and video data streams.

The data selection/queue portion of this component stores partially compressed audio and video (encoded data) in queues, and can select between queues containing compressed audio/video streams to perform the buffer input control function.

9.1 External event notification

In some circumstances, it may be useful to provide a notification to the buffer model of a state transition that was detected external to the model itself. In some cases, these events can be detected by parsing state transition messages directly from the streaming media application protocol (e.g., RTMP). In other cases, it can be inferred from out-of-band information or via a heuristic. In all cases, these state change notifications can be fed directly into the buffer model's "external state over-ride" block.

9.1.1 Event detection

There is considerable uncertainty about the visibility of critical information in streaming media streams due to the growing use of encryption. Information describing the stream (meta-data) may be completely concealed.

This clause suggests heuristics to infer stream events from measureable network conditions. There is substantial information in the TCP transport protocol, which should be exposed despite encryption on higher layers and information. Meta-data used in combination with network conditions and TCP information will increase the confidence in the inferences below. The detection approach treats the player and the stream information above TCP layer as opaque. Measurements of the TCP connection status inform the buffer model about network state and stream state.

It may be possible in some cases to characterize a particular client or set of clients with respect to signals extracted from the Transport layer and below. Such characterization could be used to provide a signal to the "external state over-ride" block in the buffer model in cases where buffer modelling would be otherwise impossible. While a specific feature matrix is for further study in this Recommendation, the following is a partial list of features that could be used in an event-detection heuristic, along with possible inferred events:

– TCP SYN from Player IP Address:
  - If the player is not actively streaming, this can indicate the start of a stream.
  - If the player is currently active, this can indicate that a new segment or chunk of the stream is being conveyed.
  - If the player is currently active and the server IP address has changed, this can indicate that a new content server is supplying the Stream.

  NOTE – Client errors, Client re-starts of a stream, and other conditions are also possible.

– TCP segments are observed out-of-order (because they are missing, or they are retransmissions):
  - Packet loss is occurring. If the network download rate is lower than the media bit rate, this could indicate that the client's buffer is depleting. If this condition is sustained, re-buffering may be occurring.

– No stream traffic throughout a waiting time, possibly accompanied by TCP FIN or RST to/from player IP Address:
  - Stream is Ending (no other TCP connections are active).

  NOTE – FIN packets alone are not sufficient. Adaptive video delivery systems may use a new TCP connection for every "chunk" of video sent, and therefore FIN packets are sent regularly. Therefore, the over-riding heuristic to detect the playing to end event is to observe the absence of stream packets throughout a waiting time, in combination with lack of evidence of retained connectivity (for the player and server(s)).

Note that the use of multiple TCP streams and servers (with multiple Server IP addresses) involved in the Stream delivery can make inference from Network state/status much more challenging.

10 Buffer configuration

The buffer configuration entity sets the thresholds for starting and stopping the video play-out. Also, the size of the buffer is configured. In order to have comparable buffer models, a set of default thresholds is provided in this Recommendation.

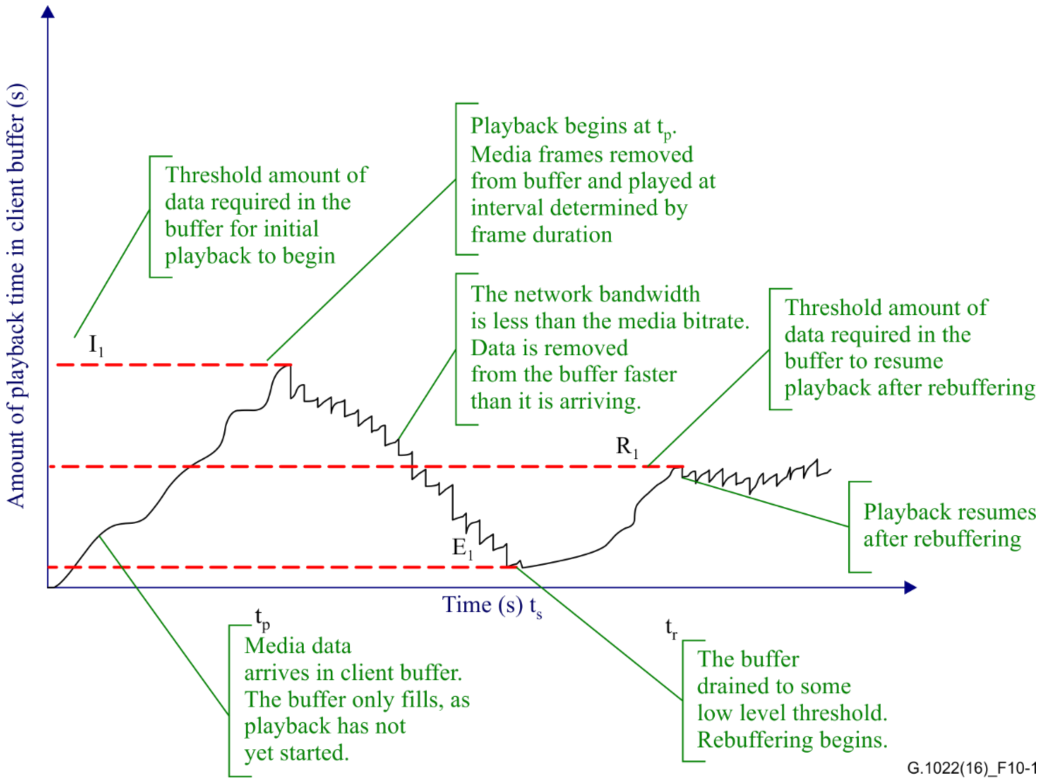

Figure 10-1 – Playout buffer fullness over time with thresholds explained

In Figure 10-1, an I_1, R_1 and E_1 are defined; furthermore, there may be multiple I_n, R_n, and E_n ($n \in \mathbb{N}$) values. Figure 10-1 above depicts how parameters may change over time. This multitude of thresholds arises from dependencies on media site, device, player version and other factors resulting in numerous configuration parameters. Thus the most general way of defining these thresholds is to have a list of I_n, R_n and E_n values where each time the model enters a re-buffering state, it uses the next R_n value from the list, and each time it enters playing state, it uses the next E_n value from the list. Such unique configuration lists can be generated for each device, media site, player version, etc. Maintaining such lists is a huge challenge, as these change frequently; therefore, this is outside of this scope of work, although the community should define reasonable default lists of I_n, R_n and E_n values that can be used as a benchmark and as a substitute when up-to-date lists are unavailable.

Instead of the given list, the behaviour could be implemented as a set of formulae. For each device, media site, player version, etc., there has to be a unique formula to calculate I_n, R_n and E_n after a state change occurs.

In the simplest case, there are $I_1 = I_n$, $R_1 = R_n$ and $E_1 = E_n$, which means the thresholds are static; in this case, only I_1, R_1 and E_1 have to be specified.

Inputs:

– Auxiliary information about: end device, player version, browser version;

– Buffer model state changes (resuming, re-buffering).

Outputs:

– Maximum buffer capacity;

– I_1, R_1 and E_1 as well as I_n, R_n and E_n after each state change.

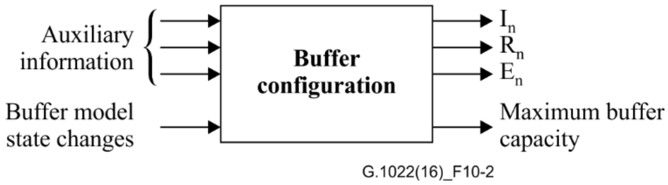

Figure 10-2 – Buffer configuration inputs and outputs

A list or formula from the existing configurations as well as a buffer size are chosen from the input information, and for each state change event, R_n or E_n, is calculated depending on whether a start resume or re-buffering event is reported from the buffer model. In some cases, it could also be possible that a new I_n is calculated and sent to the buffer model.

11 Play-out buffer

The play-out buffer model accepts frames of media from the preceding model stages, and different types of control signals. Primarily, the model calculates the number of frames present in a variable, B, and re-calculates B every time a new frame arrives.

The play-out buffer uses a set of input parameters to determine its internal control of buffer state. The input parameters include:

- I: Initial buffering threshold – transition to play state;
- R: Re-buffering threshold – transition to play state;
- E: Empty buffer threshold – transition to re-buffering state;
- O: Overflow buffer threshold – transition to exception state.

Explicit state control can also arrive from other model stages.

The buffer model re-calculation of size is also performed on a play-out control signal where the time-scale of these interrupts is determined from a play-out rate or schedule derived from Meta-data. The external control can also mask this interrupt, for example if the system enters a Pause or Stop state.

There can be multiple versions of the play-out buffer intending to serve audio and video media separately (with additional processing of both buffer states to maintain synchronization, or lip-sync).

Figure 11-1 below illustrates play-out buffer model operation. The flow is from top to bottom, with external control inputs (and output) on the right-hand side. When the buffer state changes, key variables are recorded and reported.

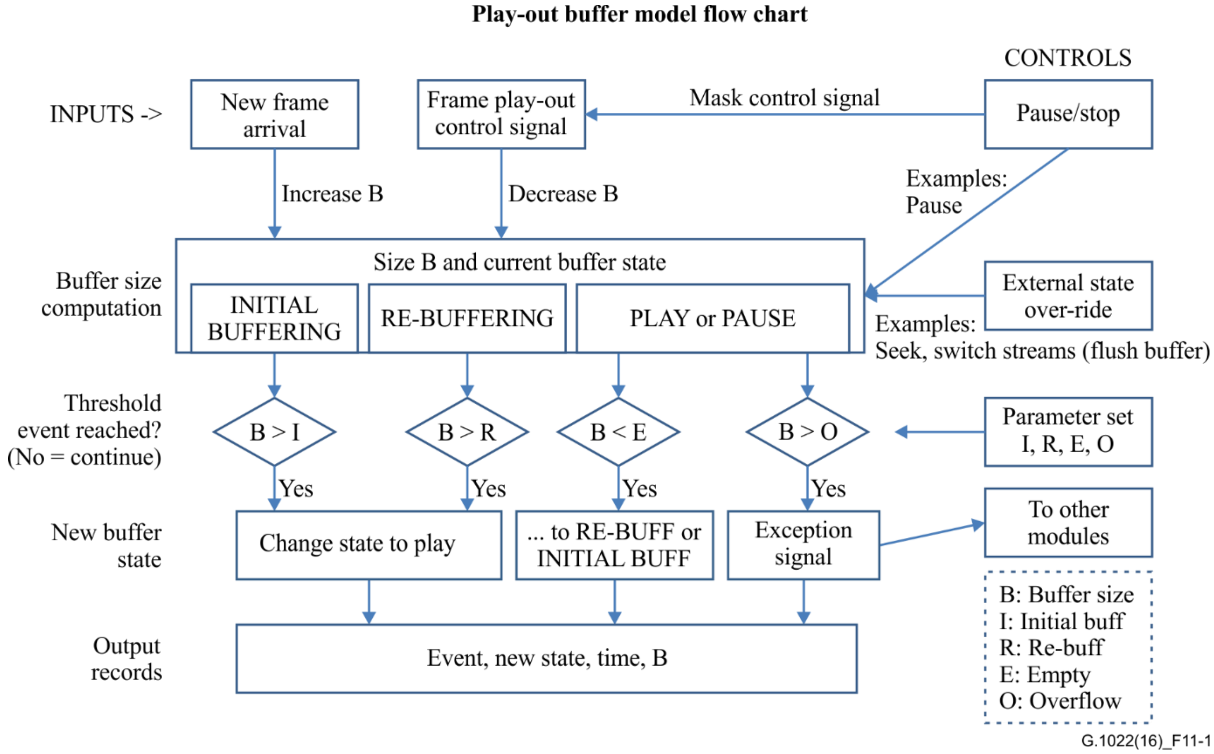

Figure 11-1 – Play-out buffer model flow chart

Beginning with inputs, frame arrival can occur at any time, while play-out interrupts are usually periodic events, and each input results in re-calculation of buffer size (occupied length of the buffer).

With each revision of B, computations are performed depending on the current buffer state, Initial buffering, re-buffering, or Play. Note that the computed buffer state has the possibility to be over-ridden by external controls, such as a pause control, which would stop play-out temporarily.

When the buffer is in initial buffering state for example, the new computed size is compared to the corresponding threshold, I, and if B exceeds I, further actions are taken. Re-buffering and Play states have other thresholds corresponding to actions (usually buffer state changes). The thresholds I, R, E, and O are variable parameters for the model and may be changed over time or in response to events occurring in the buffer (as described in clause 3.2.14, clause 10 and Annex C). Note that the overflow threshold is unique in that it results in exception notification to other buffer modules.

If none of the comparisons result in a threshold crossing, the buffer continues in its present state. Any threshold crossing that changes buffer state, shifts future comparisons to alternate thresholds. For example, B is only compared with E = empty and O = overflow in the play state.

Each state change or exception results in a new output record (stored in a log and/or communicated to downstream modules), containing the description of the event (B>R with the value of R currently in use), the new state, the time, the occupied buffer length B, and other information inferred or observed, such as a stream change.

Annex A

Video download as one single chunk within a single TCP connection

(This annex forms an integral part of this Recommendation.)

A.1 Estimating the arrival time of the video frame data at the client device

The purpose of this annex is to describe an exemplary method of estimating the arrival time of the video frame data at the client device.

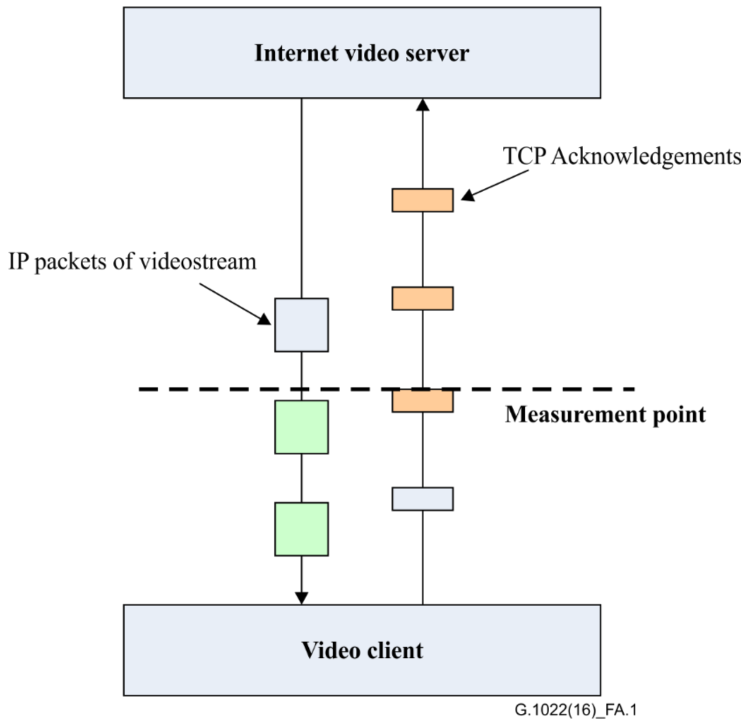

Figure A.1 – Measurement set-up for single TCP flow-based delivery

The main cause of bad quality of progressive download videos is stalling events due to delayed data reception and resulting buffer depletion times. To determine these events, it is necessary to estimate the fill level of the playout buffer and to detect depletion events. As the measurement does not have access to the user's end device, it relies on the data which can be observed at a measurement point within the network. The required information needs to be extracted from TCP segments since YouTube and other progressive download streaming services are based on HTTP/TCP transport. Therefore, the TCP segment information and the TCP payloads of the media data have to be analysed. This analysis of the TCP-based video download increases the accuracy of estimating buffer fill level based on the video timestamps encoded within the video payload of the respective TCP connection. For this extraction, byte offsets of frames in the media stream must be extracted from the media container meta-information or otherwise estimated. After determining the playout timestamp (embedded within the media payload of the TCP segment), it is compared to the observation (reception) timestamp of the corresponding TCP segment at the monitoring point. This comparison leads to an estimated buffer fill level for the observed video.

A.2 Higher performance based on not analysing each data packet

Three implementation methods have been defined for balancing the required processing power to the obtained accuracy:

1) Exact method

In the most precise "exact method", the playout time is extracted from each packet of the video flow and is compared with the timestamp of the respective TCP segment. Out of this comparison, the fill level of the playout buffer is estimated. This exact method has proven to be most precise, but leads to a high processing load. (Note – There is no processing of the elementary stream.)

2) Estimation method

The estimation method is a variation of the exact method aiming for a higher processing speed. The idea is to fully decode only the header information of the video container. The collected information out of the header includes the respective size and duration information of all blocks of the video. The estimation algorithm then calculates the fill level based on those block sizes and the observed amount of data streamed. However, timing precision deteriorates as retransmissions and packet losses occur. There is a trade-off between processing speed-up and accuracy.

3) Combined method

The combined method tries to combine the gained speed-up of the estimation method with the accuracy of the exact algorithm. This is obtained by dynamically adopting the processing mode (exact mode/estimation mode) to the experienced transport condition. Frequent transport impairments lead to buffer level reduction and low fill levels call for a more precise measurement. The decision to swap between the modes of operation is based on a single threshold value of the buffer fill level: the exact method is used only if the buffer fill level is below the critical fill level threshold.

Annex B

Considerations for network layer and container layer processing

(This annex forms an integral part of this Recommendation.)

The pre-processing details of media streams change frequently, and may be quite complex for some stream types.

Frame boundaries are not known in encrypted streams, for example, therefore approximated frames of information will be acted upon.

The following information should be captured and associated with a frame of video data that is available to the buffer model. The information transcends what is usually available to the decoder and definitely goes cross layer:

– time of transfer;
– bytes transferred;
– count of bytes transferred.

Items that potentially could be captured and transferred by the network layers:

– time of end of receipt of media frame;
– byte count of media frame;
– number of packets in transfer;
– TCP parameters of epoch: RTT avg, min, max, Retrans counts;
– RTP parameters: missing packet counts.

Within the monitored data stream for example, the media presentation description (MPD) can be identified by the clear text string "<MPD" followed by an XML structure. Its index list needs to be parsed for video segment file locations and optionally for their byte ranges within the referenced files. HTTP requests to these files can then be identified as part of the monitored video stream.

See clause 9 (on the event detector) for additional inferences based on observation of TCP protocol.

Annex C

Blackbox player measurement and buffer model parameterization

(This annex forms an integral part of this Recommendation.)

C.1 Introduction

There is considerable uncertainty about the visibility of critical information in streaming media streams due to the growing use of encryption. Information describing the stream (meta-data) may be completely concealed. At the same time, stream adaptation has become more widely adopted in some streaming services, making the concealed information even more valuable to accurate assessment.

This annex suggests a method to parameterize buffer models based on observations of the performance of actual player systems in response to measureable network conditions. The model parameters can be evaluated for their estimation accuracy by performing side-by-side tests. The accuracy of the model's prediction (when parameterized this way) may be sufficient for diagnostic measurements when deployed in a mid-path location.

C.2 Blackbox test set-up

The test approach treats the player and the stream information (above a particular protocol layer) as a blackbox. Measurements at the blackbox input and output inform the user about model parameters. Once a candidate set of model parameters is available, the next step is a verification phase where the buffer model receives the same stream as the blackbox player and the outputs are compared.

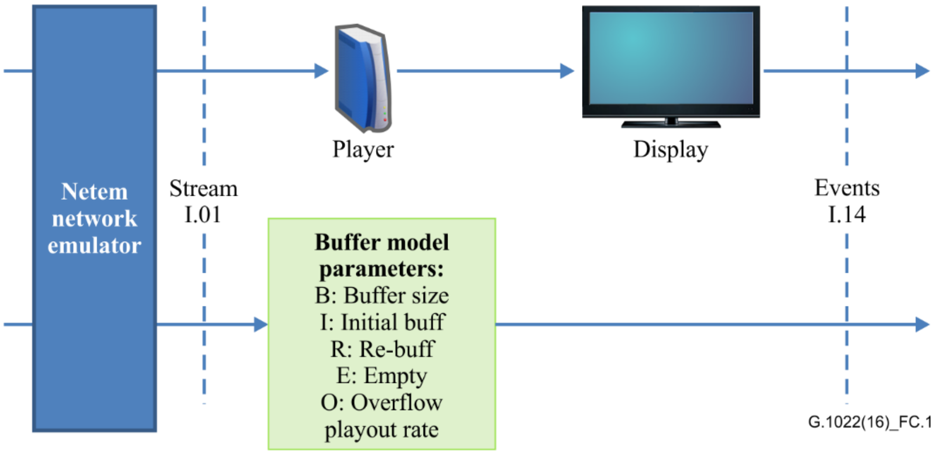

Figure C.1 – Blackbox player assessment and buffer model accuracy testing

Figure C.1 shows the test set-up, where a (bidirectional) media stream is received by the player and buffered according to an unknown design. The input stream is monitored and measured to assess some critical properties, such as the transmission bit rate over various intervals, and how those properties change over time.

The player produces video information for the display (and audio for speakers, not shown). The media presentation is monitored for events that correspond to buffer state changes inside the Player.

The intent of this testing is to discover the key parameters for the buffer inside the player. The events observed on the display and audio play-out should correspond to buffer state changes. For example, the amount of media data received from the start of the stream to the beginning of playout on the display represents the initial buffer threshold, I, in bits. It is also necessary to estimate the playout rate in bits per second, and to estimate how the playout rate changes over time. Adaptive rates and other parameters are more complicated to estimate, and require specific conditions produced during investigative testing.

The lower-half of Figure C.1 is described in clause C.4.

C.3 Investigative testing

Although it may be possible to extract some of the buffer parameters with observations of media streams, it appears to be more efficient to use a network impairment emulator (Netem in Figure C.1) to impair the stream, and measure the player response to specific conditions.

Various investigative test scenarios are possible:

1) Stream rate reduction to a known constant after playout begins. This may enable the assessment of the buffer-empty threshold, E, by estimating the buffer depletion rate.
2) Once re-buffering is achieved, the time and bits received enable the assessment of the Re-buffering threshold, R.
3) Observations of stream behaviour after re-buffering may indicate adaptive stream changes in terms of the delivered bit rates.

Although bit rates figure prominently in the above-mentioned examples, other stream attributes may offer predictive relationships with key events. For example, the characteristics of the stream burst pattern or ACK spacing may serve as alternative sources of event inference.

Since all media streams within the scope of this Recommendation will use TCP transport, it should be possible to detect packet losses prior to an observation point, and to calculate the ratio of such losses to total packets per unit of time. Metrics of the stream could follow the stream block metrics of [ITU-T Y.1540], and there may be a mapping between buffer events and packet loss expressed in a way that relates to more closely to TCP Goodput.

Testing with a true player allows the model-developers to test event detection methods, where the player interface can be used to control the Stream source and observe the resulting stream characteristics.

C.4 Verification/validation testing

Once enough of the buffer model parameters have been estimated, the events as predicted by a buffer model and a player can be compared side by side. This is the complete set-up illustrated in Figure C.1, where the buffer model and the player receive the same stream. The stream can be impaired by the network emulator, as described in the investigative testing clause. The comparison can take the form of exact event time series comparison, or comparison of metrics calculated on the events, such as the number of stalls/re-buffering events and average event duration.

C.5 Basis of comparison between predicted and actual events

The following event metrics represent a summarized basis for comparison:

– Sequence of time-stamped buffer state changes (player and model estimations).
– Count of stalls:
  • duration of stalls;
  • others, for further study.

C.6 Study questions

1. How many objectively characterized streams should be compared with model performance estimates?
2. How many types of streams should be included (number of connections, number of quality levels)?

3. What is the level of accuracy required for validation?
4. How does one compare the two sequences of time-stamped buffer state changes (one objectively collected from the client and the other from the model estimation)?
5. Which emulated impairment scenarios should be chosen for this process?
6. How can the process be automated?
7. Can open-source reference muffer models be made available in follow-up work? (Chemnitz had something at the start of their contributions.)
8. There is a large experimental variable space in media players, including the player software, TCP stack and player hardware, inside the black box. Some of the variables are visible, such as software versions comprising the player. How can a test set-up with a partially opaque black box result in reproducible results over time?
9. What can be learned from studying methods like those used in [b-ITU-T P.564]?

Appendix I

Frameless mode Annex A performance evaluation by TU Chemnitz

(This appendix does not form an integral part of this Recommendation.)

Annex A describes three implementation methods for frameless buffer modelling. The implementation operates in BMM0 and provides the three described buffer estimation algorithms. Those algorithms differ from each other in terms of processing speed up and achieved accuracy. In order to evaluate this trade-off, a field trial was carried out. The trial was conducted within the 3G and LTE network of a European operator. Human observations and traffic tracing were carried out simultaneously. The measurement probe was placed at the Gi/SGi interface of the network.

A test group watched a set of 17 YouTube videos (in all available resolutions) on laptops with mobile network access and noted down the video quality they experienced (each re-buffering event and the overall buffering time for each video). The traffic traces have been saved in PCAP file format for processing at later time.

Each buffer level estimation implementation method was fed with the same captured (packet level) traffic traces and the resulting buffer estimates were compared with the observations by the test users. The following is an overview of the performance evaluation results.

The table in the paper referenced below shows the number of re-buffering events and the re-buffering time determined by the test users compared to the processing results yielded by either the exact method, the estimation algorithm or by the combined algorithm. No differences were observed for "good case videos" (videos without stalling): all three algorithms detected no stalls, but the estimation method and the combined method required less than half of the processing time for the referenced implementation. For "bad case videos", the results of the exact and the combined method matched well with the human observations. However, the accuracy of the estimation method significantly decreased as the interval stepping increases. Hence, the combined method seems to be a good compromise between processing speed-up and accuracy. A speedup of 33% is achieved for the implementation method tested and the accuracy impairment is hardly noticeable.
Reference: [b-Eckert]

Appendix II

Relationship to Recommendation ITU-T P.1203

(This appendix does not form an integral part of this Recommendation.)

II.1 Introduction

Since ITU-T P.1203 tracks 1 and 2 are seen as taking advantage of the buffer models specified in this Recommendation, it is important to indicate how the Recommendations align, in terms of their inputs and outputs.

II.2 ITU-T P.1203 (track 1)

Figure II.1 provides the functional blocks and interfaces of [b-ITU-T P.1203] (track 1, track 2 is similar for the Buffer parameter extraction).

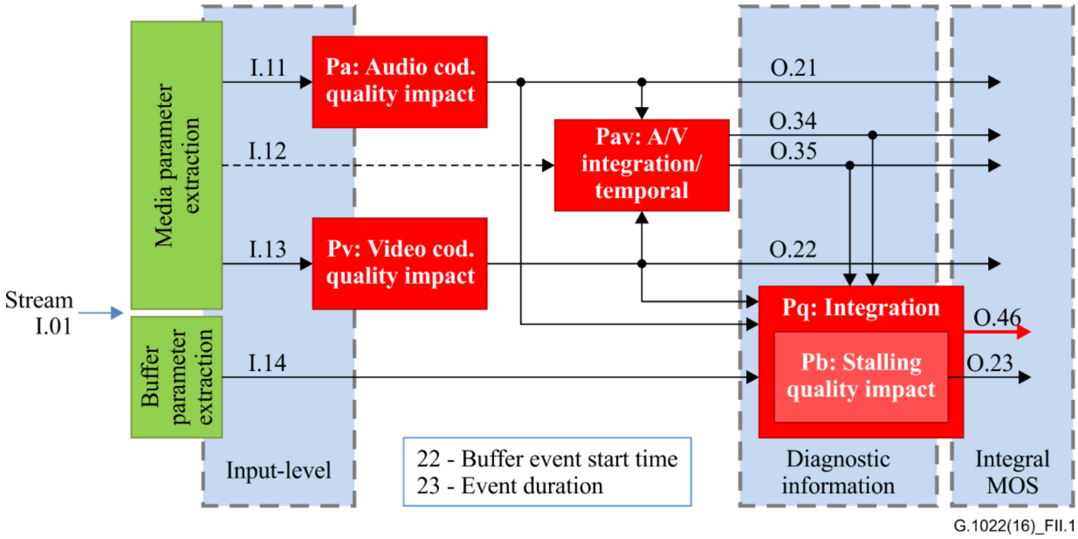

Figure II.1 – Building blocks of the ITU-T P.1203 model (Track 1)

[b-ITU-T P.1203] only considers the following state transitions:

a) Initial buffering to playing;

b) Playing to stalling;

c) Playing to end;

d) Stalling to playing.

NOTE – User-initiated state transitions are outside of the scope of this Recommendation. More specifically pausing, seeking, user-initiated quality change, user-initiated play or user-initiated end are all NOT considered.

Table II.1 – [b-ITU-T P.1203] modes of operation

Mode	Encryption	Input	Complexity
0	Encrypted media payload and media frame headers	Meta-data	Low
1	Encrypted media payload	Meta-data and frame header information	Low
2	No encryption	Meta-data and up to 2% of the media stream	Medium
3	No encryption	Meta-data and any information from the video stream	Unlimited

II.3 ITU-T G.1022 buffer mode 0

Figure II.2 shows the buffer mode 0 alignment.

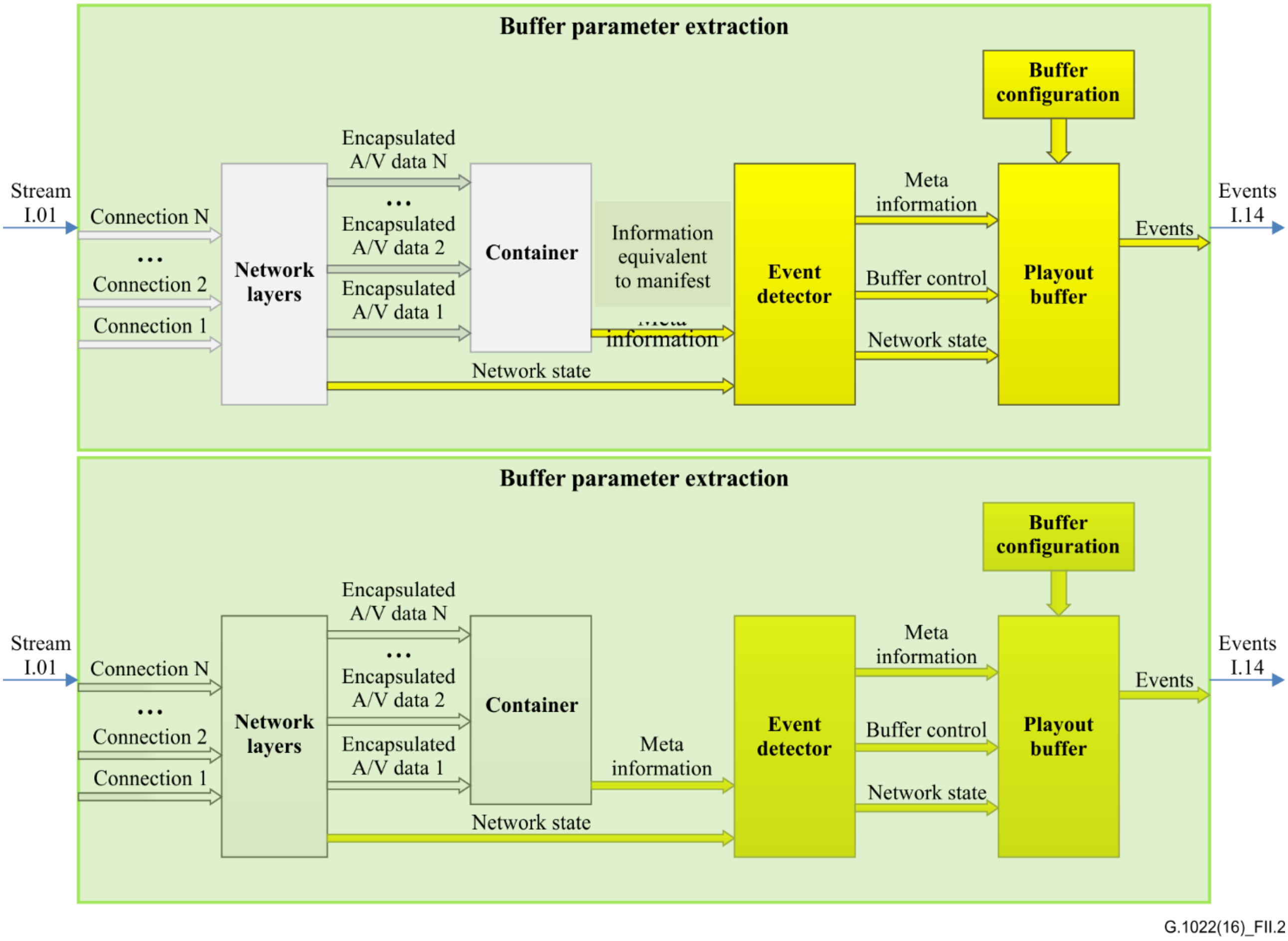

Figure II.2 – Buffer mode 0 aligned with the ITU-T P.1203 model

ITU-T G.1022 considers the following state transitions:

a) Initial buffering to playing;

b) Playing to stalling (re-buffering, or an exception state such as overflow);

d) Stalling (re-buffering) to playing;

The meta-data will be needed to supply the following transition:

c) Playing to end.

Without meta-data available, it is possible to monitor the TCP stream(s) for FIN packets, although FIN packets are not sufficient. Adaptive video delivery systems may use a new TCP connection for every "chunk" of video p to end event is to observe the absence of stream packets throughout a waiting time, in combination with evidence of retained connectivity (for the player and server(s).

It may also be possible to detect a request for a different stream resource, meaning that the playing to end event can be assumed and the new stream resource is in initial buffering. Other user actions, such as pause or seek, may have similar symptoms, but user-initiated actions are out of the scope of the Recommendation.

Bibliography

[b-ITU-T G.1021] Recommendation ITU-T G.1021 (2012), *Buffer models for development of client performance metrics*.

[b-ITU-T H.264] Recommendation ITU-T H.264 (2012) | ISO/IEC 14496-10:2014, *Advanced video coding for generic audiovisual services*.

[b-ITU T P.564] Recommendation ITU T P.564 (2007*), Conformance testing for voice over IP transmission quality assessment models*.

[b-ITU-T P.800.1] Recommendation ITU-T P.800.1 (2006), *Mean Opinion Score (MOS) terminology*.

[b-ITU-T P.1203] Recommendation ITU-T P.1203 (2016), *Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport*.

[b-ITU-T Y.1541] Recommendation ITU-T Y.1541 (2011), *Network performance objectives for IP-based services*.

[b-ETSI TS 126.234] 3GPP TS 26.234 v10.2.0 (2010), *Transparent End-to-End Packet Switched Streaming Service (PSS); Protocols and Codecs*, Release 1.

[b-ETSI TS 126.244] 3GPP TS 26.244 v10.1.0 (2011), *Transparent End-to-End Packet Switched Streaming Service (PSS); 3GPP File Format (3GP)*, Release 10.

[b-ETSI TS 126.247] 3GPP TS 26.247 (2010), *Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH) (Release 10) – 3rd Generation Partnership Project*.

[b-IETF RFC 768] IETF RFC 768 (1980), *User Datagram Protocol (UDP)*.

[b-IETF RFC 791] IETF RFC 791 (1981), *Internet Protocol*.

[b-IETF RFC 1945] IETF RFC 1945 (1996), *Hypertext Transfer Protocol Version 1.0 (HTTP/1.0)*.

[b-ISO/IEC 14496-14] ISO/IEC 14496-14 (2003), *Information technology – Coding of audio-visual objects – Part 14: MP4 file format*.

[b-Eckert] Eckert, M.; Knoll, Th. M. (2012); *An advanced network based method for Video QoE estimation based on throughput measurement*; Proceedings EuroView. http://www.euroview2012.org/fileadmin/content/euroview2012/abstracts/05_04_abstract_eckert.pdf

[b-HDS] HDS Adobe Systems (2010), *HTTP Dynamic Streaming*.

[b-HLS] HLS Apple Inc. (2009), *HTTP Live Streaming*.

[b-OIPF Spec. Vol. 2a] OIPF Specification Volume 2a (2010), *HTTP Adaptive Streaming V2.0 (AHS)*.

[b-RTMP] RTMP Adobe Systems (2009), *Real Time Messaging Protocol*.

[b-Smooth Streaming] Microsoft (2010), *Smooth Streaming*.

[b-WebM] WebM Google Inc. (2010), *WebM*.

[b-XML] XML WWW Consortium (2006), *Extensible Markup Language*.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	**Transmission systems and media, digital systems and networks**
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2016