ETSI TS 102 817 V1.1.1 (2007-09)

Technical Specification

Digital Video Broadcasting (DVB); Digital Recording Extension to Globally Executable Multimedia Home Platform (GEM)

Reference
DTS/JTC-DVB-177

Keywords
broadcasting, digital, dvb, tv, video

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by Joint Technical Committee (JTC) Broadcast of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI).

NOTE: The EBU/ETSI JTC Broadcast was established in 1990 to co-ordinate the drafting of standards in the specific field of broadcasting and related fields. Since 1995 the JTC Broadcast became a tripartite body by including in the Memorandum of Understanding also CENELEC, which is responsible for the standardization of radio and television receivers. The EBU is a professional association of broadcasting organizations whose work includes the co-ordination of its members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has active members in about 60 countries in the European broadcasting area; its headquarters is in Geneva.

European Broadcasting Union
CH-1218 GRAND SACONNEX (Geneva)
Switzerland
Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

Founded in September 1993, the DVB Project is a market-led consortium of public and private sector organizations in the television industry. Its aim is to establish the framework for the introduction of MPEG-2 based digital television services. Now comprising over 200 organizations from more than 25 countries around the world, DVB fosters market-led systems, which meet the real needs, and economic circumstances, of the consumer electronics and the broadcast industry.

Founded in 1988 by members of the cable television industry, Cable Television Laboratories is a non-profit research and development consortium that is dedicated to pursuing new cable telecommunications technologies and to helping its cable operator members integrate those advancements into their business objectives. Cable operators from around the world are members. CableLabs maintains web sites at www.cablelabs.com; www.packetcable.com; www.cablemodem.com; www.cablenet.org; and www.opencable.com.

The specifications for the following Java packages are contained in archive tam0887_stubs.zip in ts_102817v010101p0.zip which accompanies the present document.

- org.ocap.shared.dvr;
- org.ocap.shared.dvr.navigation;
- org.ocap.shared.dvr.media.

1 Scope

The present document defines a modular extension to GEM [1] which defines how the recording and playback of digital video (and audio) content is integrated with the GEM platform.

The present document is firstly intended to be used by entities writing terminal specifications and/or standards that extend a GEM terminal specification with digital video (and audio) recording and playback. Secondly it is intended for developers of GEM applications that wish to use digital video (and audio) recording and playback. Implementers should consult the publisher of specifications which reference GEM regarding conformance.

NOTE: The present document defines the interfaces visible to applications. Application developers should not assume that any related interface is available unless it is specifically listed. Terminal standards or implementations may have other interfaces present.

One of the primary goals of the present document is to maximize the common aspects concerning the integration of digital video / audio recording between MHP and the various GEM terminal specifications.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

For online referenced documents, information sufficient to identify and locate the source shall be provided. Preferably, the primary source of the referenced document should be cited, in order to ensure traceability. Furthermore, the reference should, as far as possible, remain valid for the expected life of the document. The reference shall include the method of access to the referenced document and the full network address, with the same punctuation and use of upper case and lower case letters.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[1] ETSI TS 102 819: "Digital Video Broadcasting (DVB); Globally Executable MHP (GEM)".

[2] ETSI ES 201 812: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.0.3".

[3] ETSI TS 102 812: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.1.1".

NOTE: The preceding reference is currently only available as DVB Blue Book A068 Revision 1.

[4] ETSI TS 102 816: "Digital Video Broadcasting (DVB); Personal Video Recorder (PVR)/ Personal Data Recorder (PDR) Extension to the Multimedia Home Platform".

[5] OpenCable Application Platform Specification ; OCAP Digital Video Recorder (DVR).

[6] DAVIC 1.4.1 p9: "DAVIC 1.4.1 Specification Part 9 - Information Representation".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 102 819 [1] and the following apply:

GEM recording specification: complete specification that extends the present document to create complete specification of how to integrate digital video (and audio) recording and playback with GEM terminal specifications

GEM recording terminal: terminal or other device that conforms to a GEM recording specification

normal play: play of video and / or audio content in the forwards direction at a rate of 1,0

NOTE: This includes live content, live content which has been time-shifted and the content which is the result of a scheduled recording.

recordable streams: streaming data within a piece of content which is to be recorded and played back

NOTE: For GEM terminal specifications, these include streams identified corresponding to clauses 7.2.1 ("Audio") and 7.2.2 ("Video") of GEM. GEM recording specifications may define other types of streams to be recordable streams for example subtitles or closed-captions.

recording: generic term that refers to a layman's concept of a scheduled or recorded event, and does not refer to an actual Java object

NOTE: The term covers recordings which have been requested but not yet started, recordings which are in progress, recordings which have completed (successfully or not) and recordings which failed with no data ever being recorded.

scheduled recordings: recordings where the target of the recording is specified either by time, duration and channel or by an identifier that is resolved to time, duration and channel

segmented recording: recording that is split into 1 or more pieces, i.e. segments.

NOTE: A recording can be split for various reasons, e.g. power was turned off and on, resources were lost and re-acquired.

timeline: conceptual progress of time inherent in an item of content, which may be referred to by applications and delivered by a transmitted timeline

timeshift recordings: recordings where the target of the recording is currently received video and audio content

transmitted time line: timeline included as part of a piece of content when that content is transmitted

EXAMPLE: DSMCC Normal Play Time.

trick mode or trick modes: generic term for the playback of video and/or audio at speeds other than 1,0 or directions other than forwards

NOTE: This includes fast and slow speed playback both forwards and backwards as well as pause.

trick-mode aware application: application signalled as being aware of trick-mode playback, i.e. with the **trick_mode_aware_flag** set to "1"

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in in ES 201 812 [2], TS 102 812 [3] and the following apply:

AIT	Application Information Table
API	Application Program Interface
DSMCC	Digital Storage Media Command and Control
DVB	Digital Video Broadcasting
DVR	Digital Video Recorder
GEM	Globally Executable MHP
JMF	Java Media Framework
MHP	Multimedia Home Platform
NPT	Normal Play Time
PDR	Personal Digital Recorder
PVR	Personal Video Recorder
SI	Service Information

4 Conventions

Where modifications to other documents are defined by quoting text from those other documents and describing how the present document considers that text to be changed, text that the present document adds into the original document is shown underlined and text that the present document removes from the original document is shown with strike-through.

The term "application" is used to cover both GEM applications and applications which are not themselves GEM applications but which are compliant with the GEM terminal specification implemented by a particular GEM recording terminal.

The present document uses the terminology that something "shall not be recorded" or "should not be recorded". This is a convention for the sake of brevity and in all cases, implementations are allowed to record items that shall or should "not be recorded" and discard them during playback. The term "shall not be recorded" is simply shorter than "shall not be recorded, or if recorded, shall be discarded during playback" and the latter is what is meant.

The term "while the content to be recorded is available" or similar is used a generic term due to the different ways in which GEM recording specifications may permit recordings to be specified. Where recordings are specified in terms of time and duration, it means during the duration of the recording. Where recordings are specified in terms of the contents of a field in signalling, it means while that field is present and has the correct value.

5 General considerations

5.1 Purpose

The present document is not intended, and should not be used, as a complete specification of how to integrate digital video (and audio) recording and playback with GEM terminal specifications. It is a framework upon which such a complete specification (known as a GEM recording specification) can be created.

5.2 Full conformance with the present document

To be fully conformant with the present document, it is required that GEM recording specifications be conformant with all normative requirements of one specification from each of the following two lists.

Table 1: MHP and GEM terminal specifications

Reference	Name
ES 201 812 [2]	MHP 1.0
TS 102 812 [3]	MHP 1.1
TS 102 819 [1]	GEM

Table 2: MHP and GEM recording specifications

Reference	Name
TS 102 816 [4]	PVR/PDR Extension to the Multimedia Home Platform
OpenCable Application Platform Specification ; OCAP Digital Video Recorder (DVR)	OCAP Digital Video Recorder

For avoidance of doubt, equipment which is fully conformant with the entire present document apart from the above clause is not fully conformant with the present document.

5.3 General requirements

The following requirements of [2] shall also apply to the packages defined in the present document:

- Clause 11.2.7 "Event model in DAVIC and DVB APIs".
- Clause 11.2.5 "Event listeners".
- Applications shall not define classes or interfaces in any package namespace defined in the present document. MHP terminals shall enforce this using the SecurityManager.checkPackageDefinition mechanism.

6 Recording and playback process

Implementations of the present document shall perform the following as part of their process for recording and playback.

6.1 Managing scheduled recordings

The process for managing scheduled recordings shall include the following activities:

1) Maintaining a list of recording requests which remain in the pending state (PENDING_WITH_CONFLICT_STATE or PENDING_WITHOUT_CONFLICT_STATE).

NOTE 1: GEM recording specifications may define mechanisms for automatically removing requests from this list if it appears the recording will never happen.

2) Maintaining a list of recording requests that have been successfully completed (COMPLETED_STATE), ones where recording started but failed to be successfully completed (INCOMPLETE_STATE) and ones where recording was scheduled but failed to start (FAILED_STATE).

3) Initiating the recording process for a pending recording request (in either PENDING_WITH_CONFLICT_STATE or PENDING_WITHOUT_CONFLICT_STATE) at the appropriate time, including awakening from a standby or similar power management state should this be necessary.

4) Maintaining references to recording requests that failed (FAILED_STATE) in the list of recording requests.

NOTE 2: GEM recording specifications may define mechanisms for automatically removing requests from this list based on some criteria they define.

NOTE 3: Mechanisms for resolving conflicts between recording requests (e.g. use of the tuner) are outside the scope of the present document and may be specified by GEM recording specifications.

5) Resolving ParentRecordingRequests, initially when the request is first made and at subsequent times depending on the GEM recording specification. This includes setting the initial state of the ParentRecordingRequest and changing that state when or if more of the request is resolved.

NOTE 4: The GEM recording specification is responsible for specifying the mechanism for resolving ParentRecordingRequests.

6.2 The recording process

The recording process shall include the following activities:

1) Identifying which recordable streams shall be recorded. If the specification of the recording request identifies specific streams then those are the ones which shall be recorded. If the specification of the recording request does not identify specific streams then the default streams in the piece of content shall be recorded.
When the recording is in progress and an elementary stream is added to or removed from the program, the device shall re-evaluate the streams to record as if the recording were just starting. If during this process the device determines that different streams need to be recorded it should change the streams being recorded without segmenting the recording if possible. If the device cannot change streams being recorded without segmenting the recording then it shall segment the recording as defined in clause 6.2.1.4.

NOTE 1: The definition of the default streams to be recorded is outside the scope of the present document and should be specified by GEM recording specifications.

2) Recording the identified streams, up to the limits in the recording capability of the GEM recording terminal. Where more streams of any one type should be recorded than the GEM recording terminal can record, streams shall be prioritized according to clause 11.4 of [1] (which in turn is according to clause 11.4.2.3 of [2] and [3]).

NOTE 2: Minimum capabilities for the number of streams of each type that GEM recording terminals should be able to record are outside the scope of the present document. The GEM recording specification is responsible for defining these.

3) Identifying recordable applications and recording them, the broadcast file system(s) carrying them and sufficient data to reconstruct their AIT entries. Applications with an Application recording description where the **scheduled_recording_flag** is set to "0" shall not be considered as recordable. On implementations which monitor for changes in the application signalling, when such changes be detected then the criteria for which applications are recordable shall be re-evaluated. If any applications which were recordable become not recordable or vice-versa then this shall be acted upon.

NOTE 3: A more complete definition of which applications are recordable (and which are not) is outside the scope of the present document and should be specified by GEM recording specifications.

NOTE 4: The requirements on a GEM recording terminal to monitor for dynamic data and behaviour of applications during a recording are outside the scope of the present document and should be specified by GEM recording specifications.

4) Recording sufficient information about all transmitted timelines which form part of the recording in order to enable them to be accurately reconstructed during playback.

5) Generating a media time which increments linearly at a rate of 1,0 from the beginning to the end of the recording.

6) Recording sufficient SI to identify the language associated with those recordable streams (e.g. audio) that can be specified by language - e.g. using org.davic.media.AudioLanguageControl "what SI should be recorded with a recorded service, e.g. info about languages?" added above requirement.

7) Handling the following cases relating to the loss and re-acquisition of resources during a recording:
   - the recording starts and a needed resource is not available;
   - the recording is in progress and a needed resource is lost;
   - the recording is in progress without a needed resource and the resource becomes available.
8) Handling the following cases relating to changes to elementary stream information during a recording:
   - an elementary stream with audio, video, or data is added;
   - an elementary stream with audio, video, or data is removed;
   - an elementary stream with audio, video, or data is changed, e.g. the PID changed.

6.2.1 Segmented Recordings

GEM recording terminal devices shall make a best effort to record as much of a recording as possible. If a recording is interrupted, the request shall not transition into FAILED_STATE while content which could be recorded remains available. The device shall start the recording whenever the reason for failure is removed. When a recording is interrupted based on power-cycle, or resource loss and re-acquisition the device shall create a new segment for any content recorded after the interruption. The same is true for elementary stream change if the device cannot continue the recording without stopping it and creating a new segment.

When a segmented recording is created the GEM recording terminal shall create:

- a continuous media time across all of the segments;
- an individual recording for each segment.

This allows an entire segmented recording to be played as a single recording, or each segment can be played as an individual recording.

6.2.1.1 Recording without interruption

When a recording begins at a scheduled start time and records through-out all the scheduled content without interruption it is a complete recording. The state transitions of a LeafRecordingRequest associated with a completed recording are as follows:

- PENDING_WITH_CONFLICT_STATE or PENDING_WITHOUT_CONFLICT_STATE before the recording is started. At some point before the recording starts needed resources must be acquired and the state transitioned to the PENDING_WITHOUT_CONFLICT_STATE, in order for the recording to be commence.
- IN_PROGRESS_STATE once recording begins and while the recording is on-going.
- COMPLETED_STATE when the recording duration expires.

6.2.1.2 Recording with resource interruption

When a recording starts from impulse record key press or the recording request is in the PENDING_WITH_CONFLICT_STATE and one or more resources for the recording are not available, the implementation shall execute the following steps:

1) Set the state to IN_PROGRESS_WITH_ERROR_STATE.
2) Set the value in an Exception to be returned by the LeafRecordingRequest.getFailedException method to org.ocap.shared.dvr.RecordingFailedException.INSUFFICIENT_RESOURCES.
3) No RecordedService is created and the LeafRecordingRequest.getService method SHALL return null.

When a recording is in progress in the IN_PROGRESS_STATE, or IN_PROGRESS_INSUFFICIENT_SPACE_STATE, and one or more resources for the recording are lost, the implementation shall execute the following steps:

1) Set the state to IN_PROGRESS_WITH_ERROR_STATE.

2) Stop the recording. This recording becomes the first segment in the segmented recorded service. The LeafRecordingRequest for this recording shall contain a SegmentedRecordedService where the getSegments method returns an array of services with this recording in the first, i.e. [0], location.

3) Set the value in an Exception to be returned by the LeafRecordingRequest.getFailedException method to org.ocap.shared.dvr.RecordingFailedException.INSUFFICIENT_RESOURCES.

When a recording is in progress and the state is IN_PROGRESS_WITH_ERROR and resources for the recording become available the implementation shall execute the following steps:

1) Set the state to IN_PROGRESS_INCOMPLETE_STATE.

2) Create a recorded service for the remaining recording as soon as the implementation detects the scheduled recording should be in progress.

3) Place the new recorded service in the next available location in the recorded service array returned by the getSegments method of the SegmentedRecordedService contained in the LeafRecordingRequest.

When a recording is in progress in the IN_PROGRESS_INCOMPLETE_STATE and one or more resources for the recording are lost the implementation shall execute the following steps:

1) Set the state to IN_PROGRESS_WITH_ERROR_STATE.

2) Stop the recording.

3) Set the value in an Exception to be returned by the LeafRecordingRequest.getFailedException method to org.ocap.shared.dvr.RecordingFailedException.INSUFFICIENT_RESOURCES.

If resources are removed and re-applied to a scheduled recording multiple times while the content to be recorded is available, a RecordedService shall be created and added to the SegmentedRecordedService for each time the resources became available to commence the same recording.

When a recording duration ends and the recording is in the IN_PROGRESS_WITH_ERROR state and any part of the recording request was recorded, the LeafRecordingRequest SHALL be transitioned to the INCOMPLETE_STATE. Otherwise, the LeafRecordingRequest SHALL be transitioned to the FAILED_STATE.

Typically, poor signal strength or quality is not considered loss of a resource. However, devices may segment a recording if signal or strength quality is determined unacceptable based on implementation dependent criteria. When a device segments a recording based on poor signal strength or quality it shall take action as if the associated resource, e.g. tuner, was lost. Conversely, if the signal strength or quality improves to an acceptable level based on implementation dependent criteria the device shall take action as if the associated resource was restored.

6.2.1.3 Recording with power interruption

When a scheduled recording starts when the GEM recording terminal device is powered off, but then the device is powered on while the scheduled recording is still in effect the implementation shall execute the following steps:

1) Set the state of the LeafRecordingRequest in accordance with available resources as per clause 6.2.1.2.

2) If resources are available, create a RecordedService, place it in the next available location in the SegmentedRecordedService contained in the LeafRecordingRequest and commence an associated recording.

3) If the recording completes, set the LeafRecordingRequest state to INCOMPLETE_STATE.

4) Set the Exception to be return by the LeafRecordingRequest.getFailedException method to org.ocap.shared.dvr.RecordingFailedException.POWER_INTERRUPTION.

When a scheduled recording is in-progress and the power is removed from the GEM recording terminal, but then the device is powered on and the content to be recorded has finished the implementation shall execute the following steps:

1) Set the LeafRecordingRequest state to INCOMPLETE_STATE.

2) Set the Exception to be return by the LeafRecordingRequest.getFaileException method to INSUFFICIENT_RESOURCES.

When a scheduled recording is in-progress and the power is removed from the GEM recording terminal, but then the device is powered on and the content to be recorded is still in progress the implementation shall execute the following steps:

1) If resources are available, create a RecordedService for the remaining recording as soon as the boot process completes and the implementation detects a scheduled recording should be in-progress. Add the new RecordedService to the SegmentedRecordedService referenced from the LeafRecordingRequest.

2) If the recording completes, set the LeafRecordingRequest state to INCOMPLETE_STATE.

3) Set the Exception to be return by the LeafRecordingRequest.getFailedException method to org.ocap.shared.dvr.RecordingFailedException.POWER_INTERRUPTION.

If power is removed and re-applied to the GEM recording terminal multiple times during the duration of a scheduled recording, a RecordedService shall be created for each time the power was re-applied and resources were available to commence recording. This RecordedService shall be added to the SegmentedRecordedService referenced by the LeafRecordingRequest.

6.2.1.4 Recording with elementary stream change

When a service is being recorded and an elementary stream changes (e.g. PID), the preferred behavior is for the GEM recording terminal device to follow the change without interrupting the recording. However, this is not a requirement and devices that are not capable of this behaviour are allowed to segment the recording. When an elementary stream in a service being recorded changes and the device segments the recording the device shall execute the same steps defined for loss and re-acquisition of resources in clause 6.2.1.2 Recording with resource interruption. An elementary stream change is considered a loss of a resource with an immediate re-acquisition of that resource and the device should make the gap between the recorded services as small as possible with 0 time being the goal.

6.2.1.5 Recording failure

If no part of a recording can be recorded and the LeafRecordingRequest remains in the IN_PROGRESS_WITH_ERROR_STATE throughout the duration of the recording, the device shall execute the following steps:

1) The LeafRecordingRequest is placed in the FAILED_STATE.

2) Set the value in an Exception to be returned by the LeafRecordingRequest.getFailedException method to org.ocap.shared.dvr.RecordingFailedException.INSUFFICIENT_RESOURCES.

6.3 Managing completed recordings

The process for managing completed recordings shall include the following activities:

1) maintaining with all completed recordings (COMPLETED_STATE or INCOMPLETE_STATE) the following information as long as the content is retained:

   - whether the recording is known to be complete or incomplete or whether this is unknown;
   - whether the recording is segmented and information about each segment;
   - the time and channel where the recording was made;
   - the application specific data associated with the recording.

2) possibly deleting the recording (including the entry in the list of recordings, the recorded data and any other associated information) once the expiration period is past for the leaf recording request corresponding to this recording.

NOTE: The present document is silent about the precision with which the expiration period must be respected. GEM recording specifications should specify how accurately it should be enforced by implementations.

6.4 Playback of scheduled recordings

6.4.1 Process for playback

Except when the **initiating_replay_flag** in the Application recording description of any application in the recording is set to "1", the process for playing back scheduled recordings shall include the following activities:

1) starting the playback of recordable streams;

2) starting the playback of recorded AUTOSTART applications where these form part of the recorded piece of content and all labelled parts of the application with the storage priority "critical to store" have been stored.

NOTE: Requirements on reconstructing the dynamic behaviour of recorded applications during playback are outside the scope of the present document and should be specified by GEM recording specifications.

3) When playing content which is currently being recorded, if the end of the content to be recorded is reached and recording stops, the playback must continue without interruption (but not necessarily perfectly seamlessly), regardless of any (implementation dependent) process to copy the newly recorded content from any temporary buffer to a more permanent location on the storage device.

4) A time shall be synthesized which increases linearly from the start of the recorded content to the end. This shall be used as the basis of the "time base time" and "media time" as defined by JMF. No relationship is required between this time and any time that forms part of the recorded content such as MPEG PCR, STC or DSMCC NPT.

5) When playing an entire recording with multiple segments, the segments shall be treated as a single recording. The segments shall be played in the order they appear in the array returned from the SegmentedRecordedService.getSegments method, and a time shall be synthesized for all of the segments as described in rule number 4 above. When playing across a gap between two segments the (implementation dependent) time of the gap should be minimized as much as possible with 0 gap time the goal.

6) When playing a single segment from a recording that includes multiple segments the single segment shall be treated as an entire recording playback. The beginning and end of the segment shall be the beginning and end of the playback.

7) For all transmitted time lines which form part of the original recording, reconstruct each time line when the current media time is in the range for which that time line is valid.

Applications where labelled parts of the application with the storage property "critical to store" have not been stored shall be treated as if none of the application has been stored. They shall not appear to be present.

When the **initiating_replay_flag** in the Application recording description of any application in the recording is set to "1", the GEM recording terminal shall launch the first auto-start application in the recording which has **initiating_replay_flag** set to "1". The GEM recording terminal shall not initiate the playback of the recordable streams since this is the responsibility of that application.

6.4.2 Events during playback

During the playback of content recorded as the result of a scheduled recording, the following behaviour shall be supported:

- The events generated when playback reaches the start and end of the recorded content shall be specified by the GEM recording specification.

6.5 Timeshift

6.5.1 The recording process

The process for timeshift recording shall include the following activities:

1) Identifying which recordable streams are to be recorded and recording them.

NOTE 1: The definitions of which streams are to be recorded in timeshift recording is outside the scope of the present document and should be specified by GEM recording specifications.

2) Identifying recordable applications signalled with the content being recorded at the start of the recording and recording at least those applications and sufficient data to reconstruct their AIT entries.

NOTE 2: The definition of which applications are recordable in timeshift (and which are not) is outside the scope of the present document except as defined by the time_shift_flag. A more complete definition should be specified by GEM recording specifications.

NOTE 3: The requirements on a GEM recording terminal to monitor for dynamic data and behaviour of applications during a timeshift recording are outside the scope of the present document and should be specified by GEM recording specifications.

3) Identifying the transmitted time lines which form part of the content being recorded and recording sufficient information about them to enable them to be accurately reconstructed during playback.

4) Managing the timeshift buffer such that when the buffer is full, recording continues with the oldest remaining content in the buffer being progressively over-written.

6.5.2 Playback

Playback of content recorded in timeshift mode requires a time-shift buffer to be associated with a JMF player or service context. The definition of how and when this association is made is outside the scope of the present document and should be specified by GEM recording specifications.

During the playback of content recorded in timeshift mode, the behaviour shall be as defined in clause 6.4 with the following modifications:

1) The events generated when playback reaches the start and end of the recorded content shall be as specified by table 3.

Table 3: Events during timeshift playback and resulting behaviour

Event	Behaviour	Java Event
Fast forward to end of timeshift buffer (i.e. the point where newly recorded content is being written)	Switch to playback with rate=1.0 within the buffered recording or live without destroying the contents of the timeshift buffer.	org.ocap.shared.media.EndOfContentEvent + org.ocap.shared.media.EnteringLiveModeEvent
Rewind to the beginning of the timeshift buffer (i.e. the point where recorded content is being over-written)	Switch to normal play playback with rate=1.0 within the buffered recording, from the beginning of the content in the timeshift buffer.	org.ocap.shared.media.BeginningOfContentEvent
Play to end of timeshift buffer (i.e. bad signal reception causes a data underflow in the timeshift buffer)	Either do nothing (i.e. continue playback with rate=1.0 within the buffered recording) or switch to live without destroying the contents of the timeshift buffer.	If the behavior is "switch to live without destroying the contents of the timeshift buffer" then a org.ocap.shared.media.EnteringLiveModeEvent shall be generated. Otherwise no event shall be generated.
Pause or playback at less than rate=1.0 until timeshift buffer is "full" and the content at the point where pause was made or the playback point is about to be over-written.	Switch to playback with rate=1.0 from pause / playback point.	org.ocap.shared.media.BeginningOfContentEvent

2) For all transmitted timelines which are valid in the time shift buffer, reconstruct each time line when the current media time is in the range for which that time line is valid.

3) The media time for each location in the time shift buffer shall be the value assigned to that location at the point it was recorded. There shall be no discontinuities in media time within the buffered recording including the "head" of the buffer where the content is the same as the currently received content.

4) Setting the media time to a value corresponding to POSITIVE_INFINITY shall set the playback location to the current record point if the recording is still on-going, or, if not, to the end of the recording.

5) If the playback location is the same as the record point (i.e. the live point is being played back), the content shall be displayed with a delay of zero relative to the original broadcast content. Implementations which can display the recorded content with a delay of zero relative to live may display the recorded content. Other implementations must display the original broadcast content.

7 Recording and playback APIs

7.1 Recording and recording management

7.1.1 Overview (informative)

The recording APIs enable applications to perform recordings in 2 different ways:

1) schedule a recording to be performed at some point in the future;

2) record broadcast events in real-time. This may also include the portion of the broadcast event that is in a time-shift buffer.

The implementation maintains a database of recordings, represented by the RecordingManager singleton. Applications create recordings by calling the record() method of the RecordingManager object. This operation in effect creates an entry in the recording database maintained by the implementation. These entries are represented by instances of RecordingRequest. Applications may associate application specific data with RecordingRequests using the addAppData() method. This application specific data could be event descriptions in an application private format or a key into some database of information maintained by the application.

When a recording is created, applications specify a set of recording properties. One of these is an expiration period measured from when the recording is made. As defined in clause 6.3, once this expiration period has passed, the implementation will delete the recording.

Applications may acquire a set of recordings represented by a RecordingList. Applications that wish to limit the set of recordings in a RecordingList can express this by means of a RecordingListFilter. The present document defines some filters which are available to applications. GEM recording specifications may define their own filters. Applications may create their own filters. Filters can be cascaded so that the output of one filter is used as the input to a subsequent filter.

NOTE: GEM Recording specifications may define rules governing the access of recordings to applications. Where this is done, the listing API will not return recordings which the calling application is not allowed to access as determined by these rules.

When data starts being recorded for a recording, a RecordedService is created. RecordedService extends the JavaTV service interface and forms the link between the recording API and the playback API.

7.1.2 Details

The org.ocap.shared.dvr and the org.ocap.shared.dvr.navigation package shall be supported.

In ServiceContextRecordingSpec, GEM recording specifications may make mandatory the otherwise optional feature of storing and recording the contents of the time-shift buffer when the startTime is in the past.

7.2 Playback

7.2.1 Overview (informative)

Playback of a recording can be performed in two ways depending on whether or not it is desired to start any applications which may form part of the recording.

1) Calling the select(Service) method of a ServiceContext passing in the RecordedService to be played will select all the service components in the service including any autostart applications which have been recorded.

2) Calling the getMediaLocator() method on a RecordedService and then passing the result to javax.media.Manager.createPlayer(MediaLocator). Once a Player has been obtained, start() or syncStart() will start playback. This will only playback the recordable streams within the recording and not any applications.

Once playback has started, some of the ways of controlling it include the following:

1) use of the JMF controls returned from Player.getControl(String) or Player.getControls() for features like audio language selection and video scaling;

2) use of Player.setRate(float) to control the speed of playback including changing between normal speed play, fast forwards, fast reverse and pause.

7.2.2 Details

The org.ocap.shared.media package shall be supported. JMF players presenting content where trick modes can be used shall support FrameControl from this package. JMF players presenting the contents of a timeshift buffer shall support the TimeshiftControl control from this package.

The following extensions shall apply to the APIs required by [1]:

1) The method javax.tv.service.selection.ServiceContext.select(Service) shall accept instances of RecordedService as being valid inputs. When called with such an instance, the recorded content of that RecordedService shall be selected in the ServiceContext specified.

2) The method javax.media.Manager.createPlayer(MediaLocator) shall accept the mediaLocators returned by RecordedService.getMediaLocator as being valid inputs and return a JMF Player. When such a player enters the started state, the recordable streams of that RecordedService shall be presented.

3) When a RecordedService is successfully selected in a ServiceContext, the AppsDatabase shall be populated from the set of applications recorded as part of that RecordedService as if those applications were transmitted live.

NOTE: AppsDatabase should be updated to the extent that the GEM recording specification requires monitoring and reconstructing changes in the application signalling.

4) During playback of recordable streams (both playback of a scheduled recording and timeshift recording), when the rate of playback changes, a javax.media.RateChangeEvent shall be sent to all applications with ControllerListeners registered on a JMF player for that content.

5) Calls to the setRate method of a JMF Player shall control the speed of playback including changing between normal speed play, fast forwards, fast reverse and pause.

6) Calls to the method Player.setMediaTime shall attempt to start presentation of content as close as possible to the specified media time except when passed a media time returned by MediaTimeFactoryControl. SetTimeApproximations when the semantics defined by that method shall be observed.

7) When a RecordedService is being played back, calls to the method ServiceDomain.attach(..) with the Locator returned by RecordedService.getLocator shall work as specified and provide access to the broadcast file system in the recording. The present document does not define requirements for access to broadcast file systems in a recording when that recording is not being played back however such requirements may be defined by GEM recording specifications.

When JMF players are presenting the recordable streams of a RecordedService or the contents of a timeshift buffer, at a minimum the following JMF controls (defined by [1] or the present document or as explicitly specified) shall be supported:

- org.davic.media.AudioLanguageControl;
- org.davic.media.FreezeControl;
- javax.tv.media.MediaSelectControl;
- org.ocap.shared.media.TimeLineControl;

- org.ocap.shared.media.MediaTimeFactoryControl;
- org.davic.media.MediaTimeEventControl as defined in DAVIC 1.4.1 p9 [6].

NOTE: GEM recording specifications may require additional JMF controls to be supported for RecordedServices or the contents of a timeshift buffer. Different sets of JMF controls may be specified for these two cases.

The interface org.dvb.service.selection.DvbServiceContext from [3] shall be supported. All ServiceContext instances shall be instances of DvbServiceContext. The method DvbServiceContext.getNetworkInterface shall return a special NetworkInterface as detailed below if all of the following apply:

- the GEM recording terminal supports recording and playback of streams of MPEG-2 private sections including re-constructing their timing during playback;
- the DvbServiceContext is in the presenting state;
- either the selected service is a RecordedService or the selected service has a time-shift buffer associated with it.

If any of these do not apply then this method shall return null. Such special NetworkInterfaces shall have the following characteristics:

- the method getCurrentTransportStream shall return a TransportStream object which, when passed to SectionFilterGroup.attach, shall give access to the streams of MPEG-2 private sections in the recording;
- the method listAccessibleTransportStreams shall return an array containing only the same TransportStream as returned by getCurrentTransportStream;
- no NetworkInterfaceEvents shall be generated;
- the locator returned by getLocator and the return value of the getDeliverySystemType methods are implementation dependent;
- it shall not be .equals to any NetworkInterface returned by NetworkInterfaceManager.getNetworkInterface;
- attempts to reserve it shall fail with a NoFreeInterfaceException.

7.3 Other APIs

7.3.1 Versioning

The properties listed in table 4 and table 5 shall be included in the property set of the java.lang.System class. Thus these properties can be retrieved using java.lang.System.getProperty(). Since this API returns a string, numeric return values shall be encoded as defined by java.lang.Integer.toString(int).

Table 4: System properties for version interrogation

Property	Semantics	Possible values	Example
gem.recording.version.major	Major version number of the version of the present document supported.	Non-negative integer value	"1"
gem.recording.version.minor	Minor version number of the version of the present document supported.	Non-negative integer value	"0"
gem.recording.version.micro	Micro version number of the version of the present document supported.	Non-negative integer value	"0"

The values of these properties that shall be returned in implementations of the present document are defined in clause 12.1 System constants.

7.4 Permissions

7.4.1 Unsigned applications

From RecordingPermission, only RecordingPermission ("read", "own") shall be granted to unsigned applications.

7.4.2 Signed applications

Signed applications by default shall have the same permissions as unsigned applications. Signed applications can additionally request to get more permissions. These permissions are requested using the permission request file from clause 12.6 of [2] or [3]. This clause defines the mapping from the items in the permission request file to the Java Permissions that may be granted by the MHP terminal in response to the request.

When the permission request file requests the permission to create a recording request and this is granted, a RecordingPermission ("create", "own") is created.

When the permission request file requests the permission to modify a recording request and this is granted, a RecordingPermission ("modify", "own") is created.

When the permission request file requests the permission to delete a recording request and this is granted, a RecordingPermission ("delete", "own") is created.

When the permission request file requests the permission to cancel a recording request and this is granted, a RecordingPermission ("cancel", "own") is created.

GEM recording specifications may define a mechanism for permitting applications to have RecordingPermission instances whose action string is "*" but this is not required.

8 Application signalling

8.1 Application recording description

GEM recording specifications shall define an application recording description sufficient to derive the following:

Table 5: Application recording description

Function	Type
`scheduled_recording__flag`	boolean
`trick_mode_aware_flag`	boolean
`time_shift_flag`	boolean
`initiating_replay_flag`	boolean
`label_count`	unsigned integer
`for(i=0;i<N0;i++){`	
`label_length`	8 bit unsigned integer
`for(j=0;i<N1;i++){`	
`label_char`	8 bit unsigned integer
`}`	
`storage_properties`	2 bit unsigned integer
`}`	

scheduled_recording_flag: this single bit flag, when set to "1", indicates that the application is appropriate to record when the service in which it is signalled is recorded by a scheduled recording. When set to "0", it indicates that the application is inappropriate to record by a scheduled recording. Examples of why an application would be inappropriate to record include the application not having been tested in a PDR environment or that the application is closely related to the time of transmission and would be meaningless to the end-user if played back from a recording (e.g. an application tied to a live event).

trick_mode_aware_flag: this single bit flag, if set to "1", indicates that the application is trick-mode aware. If set to "0", the application is not aware of trick-modes.

time_shift_flag: this single bit flag, when set to "1", indicates that the application is appropriate to record when the service in which it is signalled is recorded in time-shift recording mode. When set to "0", it indicates that the application is inappropriate to record in time-shift recording mode.

initiating_replay_flag: this single bit flag, if set to "1", it indicates that the GEM recording terminal shall not initiate the playback of the recordable streams in the same recording as the application. The application is responsible for starting this playback. If set to "0", the implementation shall initiate this playback in parallel with starting the application as would conventionally be the case. This flag shall only be considered when a RecordedService is initially selected. After this time, the value of this flag shall be ignored.

label_count: this 8-bit field identifies the number of labels that have been used.

label_length: this 8-bit field identifies the number of bytes in the label.

label_char: these 8-bit fields carry an array of bytes that label a part of the application within its transport protocol.

NOTE: The present document does not define which parts of applications can be labelled or the form of the label (if any). Labelling can be done at the level of files or groups of files or some lower level construct specific to the protocol used to transport the application.

storage_properties: A field indicating the importance of storing the labelled part of the application. Values for this field are defined as follows:

0 should not be stored.

1 critical to store.

2 optional to store.

3 reserved.

GEM recording specifications that include the MHP definition of the GEM "Application Signalling" functional equivalent shall fulfil this requirement by supporting the application recording descriptor defined in annex A.

9 Application model

9.1 Application lifecycle and trick-mode playback

When playback of recorded content has been initiated by selecting a RecordedService, the application model and lifecycle shall be modified as follows:

1) Running applications not explicitly signalled as trick-mode aware (the trick_mode_aware_flag in the Application recording description is set to "1") shall be killed when playback changes from normal to a trick-mode. Applications explicitly signalled as trick-mode aware shall continue running.

2) When playback leaves trick-mode and returns to normal, the GEM recording terminal shall evaluate the application signalling for that point in the content and start or stop applications as necessary. Applications which are started shall be started as if the end-user had just changed to a broadcast of the content concerned.

NOTE: GEM recording specifications may permit delays in re-starting applications after the return to normal play if this is believed to improve the end-user experience, for example during repeated cycles of fast-forward / play / fast-forward / play.

Transitioning from live playback to time-shifted playback shall not automatically kill any MHP applications while the time shifted content is being played from the end of the time-shift-buffer, (i.e. the point where newly recorded content is being written).

10 Security

10.1 Introduction (informative)

The present document includes 2 security models governing the ability of applications to operate on recording requests and completed recordings.

1) One model is based on associating attributes with MHP/OCAP applications. These attributes are expressed as Java Permission classes. Certain method calls are protected and throw a SecurityException if applications without specified Permissions call them. This model is independent of the details of the RecordingRequest concerned.

2) The second model is based on associating security attributes with individual recording requests. These attributes determine which applications may perform which operations on that recording request. The present document is silent about the mechanism by which these attributes are associated with a recording request.

In the normal case, an application's ability to use the features provided by the present document is governed by the intersection of both sets of attributes. Operations on a RecordingRequest will fail unless the calling application has the necessary Java Permission for the operation and the attributes associated with the RecordingRequest being operated upon permit the calling application to perform that operation. GEM recording specifications may define a mechanism for highly trusted applications to obtain a Permission which enables them to bypass the second model and have the right to perform all operations on all RecordingRequests.

10.2 Permission request file

The DTD of the permission request file defined by the GEM terminal specification shall include at least the following element and associated attributes:

```
<!ELEMENT recordingpermission EMPTY>
<!ATTLIST recordingpermission
    create (true|false) "false"
    modify (true|false) "false"
    delete (true|false) "false"
    cancel (true|false) "false"
>
```

11 Minimum receiver requirements

The following requirements shall apply to all GEM recording terminals:

1) At least the following rates of variable speed playback shall be supported: -16x, -8x, -4x, -2x, -1x, 0, 0.5x, 1x, 2x, 4x, 8x, 16x.

NOTE: For the -1x rate, it is allowed to only display i-frames.

12 Registry of constants

12.1 System constants

Table 6 defines the values for the system properties identifying the version of the present document supported by an implementation.

Table 6: System Property Values

Field	Value
Major	1
Minor	0
Micro	2

Annex A (normative): Application recording description

GEM recording specifications that include the MHP definition of the GEM "Application Signalling" functional equivalent shall fulfil this requirement by supporting the application recording descriptor defined as follows.

The application recording descriptor can be signalled in the application descriptor loop of the AIT. This descriptor contains extra information on application life cycle indicating in particular if an application is appropriate to use in conditions of trick-mode playback. It indicates whether this application shall or shall not be recorded, when a program, along with which this application is signalled, is recorded. It provides a means to specify the locations of data resources that shall be recorded along with the application, as well as the labels of the object carousel modules of the application that shall, should or should not be recorded.

Table A.1: application recording descriptor syntax

Syntax	No. of bits	Identifier	Comments / Value
application_recording_descriptor (){			
descriptor_tag	8	uimsbf	6
descriptor_length	8	uimsbf	
scheduled_recording_flag	1	bslbf	
trick_mode_aware_flag	1	bslbf	
time_shift_flag	1	bslbf	
dynamic_flag	1	bslbf	
av_synced_flag	1	bslbf	
initiating_replay_flag	1	bslbf	
reserved	2	bslbf	
label_count	8	uimsbf	N0
for(i=0;i<N0;i++){			
label_length	8	uimsbf	N1
for(j=O; j<N1; j++) {			
label_char	8	uimsbf	
}			
storage_properties	2	uimsbf	
reserved	6		
}			
component_tag_list_length	8	uimsbf	N2
for(i=0;i<N2;i++){			
component_tag	8	uimsbf	
}			
private_length	8	uimsbf	N3
for(i=0;i<N3;i++){			
private	8	uimsbf	
}			
for(i=0;i<N4;i++){			
reserved_future_use	8	uimsbf	
}			
}			

The semantics of the fields defined in this descriptor shall be as defined in clause 8.1 unless otherwise specified in this clause.

descriptor_tag: this 8 bit integer with value 0x06 identifies this descriptor.

dynamic_flag: this flag indicates whether the application relies on the use of dynamic data during its execution. When set to 1, it indicates that the application relies on the presence of files (either code or data) or application signalling (e.g. application control code) which change during the lifetime of the piece of content.

NOTE 1: The present document does not define behaviour for GEM recording terminals that is conditional upon the value of this flag. GEM terminal specifications may use this flag in their determination of whether or not an application is recordable.

av_synced_flag: this flag indicates whether the application requires use of the trigger events. If set to "1", this is required.

NOTE 2: The present document does not define behaviour for GEM recording terminals that is conditional upon the value of this flag. GEM terminal specifications may use this flag in their determination of whether or not an application is recordable.

label_char: these 8-bit fields carry an array of bytes that are a module label. This label matches a label on one or more modules carried by Label descriptors in the userInfo fields of the moduleInfo structure of DIIs as defined by clause B.2.2.4.1 of [2] and [3].

component_tag_list_length: this integer specifies the length in number of bytes of the list of component tags.

component_tag: this field identifies a service component that delivers data that is required by the application at playback time and that shall be recorded along with the application. Components carrying recordable streams need only be listed if components other than the default should be recorded. Components that are streams carrying DSMCC object carousels or MHP application information tables should not be listed and shall be silently ignored if they are listed. Except for these, components that are streams carrying MPEG-2 private sections should be listed if their recording is desirable.

private: these bytes may be used for private extensions.

reserved_future_use: these reserved bytes may be used for future DVB extensions.

Annex B (informative): Responsibilities of GEM Recording Specifications

B.1 Required responsibilities

The following is a list of items identified in the present document as being outside the scope of the present document and which GEM recording specifications must specify.

1) Which types of stream are to be considered as "recordable streams". Stream types corresponding to clauses 7.2.1 ("Audio") and 7.2.2 ("Video") of GEM in the GEM terminal specification on which the GEM recording specification is based must be considered as "recordable streams".

2) Minimum capabilities for the number of steams (or number of streams of each type) that a GEM recording terminal must be able to record.

3) The definition of which applications are recordable in both scheduled and timeshift recording (which need not be the same).

4) Requirements on a GEM recording terminal to monitor for dynamic data (in the DSMCC object carousel or GEM functional equivalent)during scheduled and timeshift recording (which need not be the same).

5) Requirements on a GEM recording terminal to monitor for GEM triggers or DSMCC stream events during scheduled and timeshift recording (which need not be the same).

6) Requirements on a GEM recording terminal to monitor for dynamic application signalling during scheduled and timeshift recording (which need not be the same).

7) Requirements on reconstructing the timing of dynamic data, triggers / stream events and dynamic application signalling during playback of scheduled and timeshift recordings (which need not be the same). GEM recording specifications must define requirements for both normal speed playback and trick mode playback.

8) How accurately the expiration period should be enforced by implementations.

9) The definition of at least one protocol for transmitted time lines.

10) The conditions when a JMF player or service context has a time-shift buffer attached.

11) Requirements on the size of application specific private data which it must be possible to associate with a single key without an IllegalArgumentException being thrown.

12) Requirements on the number of entries of application specific private data which it must be possible to associate with a single RecordingRequest without a NoMoreDataEntriesException being thrown.

13) A mechanism to associate security attributes with individual recording requests and a mapping from that mechanism to the language in each of the methods in the following table relating to "RecordingRequest specific security attributes".

Table B.1: Methods with dependencies on recording request specific security attributes

Method
RecordingManager.addRecordingChangedListener(RecordingListListener)
RecordingManager.getEntries()
RecordingManager.getEntries(RecordingListFilter)
RecordingManager.getRecordingRequest(int)
RecordingRequest.addAppData(int,java.io.Serializable)
RecordingRequest.removeAppData(int)
RecordingRequest.setRecordingProperties(RecordingSpec)
RecordingRequest.cancel()
RecordingRequest.stop()
RecordingRequest.delete()
RecordingManager.record()
LeafRecordingRequest.getService()
RecordedService.setMediaTime()

14) The mechanism for resolving parent recording requests including setting the initial state of a parent recording request.

15) The events generated during playback when the start and end of a recording a reached.

B.2 Optional responsibilities

The following is a list of items identified in the present document as being outside the scope of the present document and which GEM recording specifications may specify.

1) Mechanisms for controlling the extent to which one application can read or modify scheduled recordings and completed recordings made by another application.

2) Mechanisms for resolving conflicts between requested recordings (e.g. use of the tuner).

3) Sub-classes of RecordingListFilter to filter the list of recordings in ways not supported by the present document.

4) Rules governing which recordings an application can access.

5) Additional JMF controls to be supported for RecordedServices or the contents of a timeshift buffer. Different sets of JMF controls may be specified for these two cases.

6) Delays in re-starting applications after the return to normal play if this is believed to improve the end-user experience, for example when repeated cycles of fast-forward / play / fast-forward / play.

7) A mechanism to permit highly trusted applications to obtain instances of RecordingPermission whose action parameter is "*".

8) Whether the optional behaviour defined in the class description of ServiceContextRecordingSpec, (where the contents of the time-shift buffer are stored when the startTime parameter is in the past), becomes mandatory.

9) Whether there are any requirements on ServiceRecordingSpec or LocatorRecordingSpec to support including content in the past as part of a scheduled recording if that content has been recorded for some other reason like time-shift.

10) Whether there are any requirements on ServiceDomain.attach to provide access to the transport mechanisms of a recording when that recording is not being played back.

11) Any requirements on power management including any particular modes like a low-power mode and any requirements to transition to / from those mode(s) in response to scheduled recordings.

12) Whether there are any requirements to combine pieces of a scheduled recording interrupted by power loss into a single construct (e.g. a sub-class of RecordingRequest) enabling them to be played back and otherwise manipulated as a single entity.

13) Mechanisms for automatically removing requests from the list of recordings in a pending state if it appears the recording will never happen.

14) Mechanisms for automatically removing requests from the list of recordings in a failed state based on some criteria they define.

15) Any requirements to re-resolve ParentRecordingRequests after the request has first been made and to update the state accordingly.

Annex C (normative): External references; errata, clarifications and exemptions

C.1 Java Media Framework

C.1.1 javax.media.Clock

In the present document, the following text in the specification for this class shall be clarified as described below.

The transformation that a `Clock` defines on a `TimeBase` is defined by three parameters: rate, *media start-time* (mst), and *time-base start-time* (tbst). Given a *time-base time* (tbt), the *media time* (mt) can be calculated using the following transformation:

```
mt = mst + (tbt - tbst)*rate
```

The rate is simply a scale factor that is applied to the `TimeBase`. For example, a rate of 2.0 indicates that the `Clock` will run at twice the rate of its `TimeBase`. Similarly, a negative rate indicates that the `Clock` runs in the opposite direction of its `TimeBase`.

The *time-base start-time* and the *media start-time* define a common point in time at which the `Clock` and the `TimeBase` are synchronized.

Each successful rate change shall be considered to be "a common point in time at which the `Clock` and the `TimeBase` are synchronized" as defined above. The values of the Clock and TimeBase at the common point shall be the values at the instant of the rate change.

History

Document history		
V1.1.1	September 2007	Publication