ETSI TR 102 397-12 V1.1.1 (2005-08)

Technical Report

**Open Service Access (OSA);
Mapping of Parlay X Web Services to Parlay/OSA APIs;
Part 12: Multimedia Conference Mapping**

Reference

DTR/TISPAN-01021-12-OSA

Keywords

API, OSA, service

Contents

Intellectual Property Rights

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN).

The present document is part 12 of a multi-part deliverable covering Open Service Access (OSA); Mapping of Parlay X Web Services to Parlay/OSA APIs, as identified below:

Part 1: "Common Mapping";

Part 2: "Third Party Call Mapping";

Part 3: "Call Notification Mapping";

Part 4: "Short Messaging Mapping";

Part 5: "Multimedia Messaging Mapping";

Part 6: "Payment Mapping";

Part 7: "Account Management Mapping";

Part 8: "Terminal Status Mapping";

Part 9: "Terminal Location Mapping";

Part 10: "Call Handling Mapping";

Part 11: "Audio Call Mapping";

Part 12: "Multimedia Conference Mapping";

Part 14: "Presence Mapping";

NOTE: Part 13 has not been provided as there is currently no defined mapping between ES 202 391-13 [4] and the Parlay/OSA APIs. If a mapping is developed, it will become part 13 of this series.

The present document has been defined jointly between ETSI, The Parlay Group (http://www.parlay.org) and the 3GPP.

1 Scope

The Parlay X Web Services provide powerful yet simple, highly abstracted, imaginative, telecommunications functions that application developers and the IT community can both quickly comprehend and use to generate new, innovative applications.

The Open Service Access (OSA) specifications define an architecture that enables application developers to make use of network functionality through an open standardized interface, i.e. the Parlay/OSA APIs.

The present document specifies the mapping of the Parlay X Multimedia Conference Web Service to the Multi-Party Call Control and Multi-Media Call Control Service Capability Features (SCFs).

2 References

For the purposes of this Technical (TR), the following references apply:

[1] ETSI TR 121 905: "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Vocabulary for 3GPP Specifications (3GPP TR 21.905)".

[2] W3C Recommendation (2 May 2001): "XML Schema Part 2: Datatypes".

NOTE: Available at http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.

[3] ETSI TR 102 397-1: "Open Service Access (OSA); Mapping of Parlay X Web Services to Parlay/OSA APIs; Part 1: Common Mapping".

[4] ETSI ES 202 391-13: "Open Service Access (OSA); Parlay X Web Services; Part 13: Address List Management".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 102 397-1 [3] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 102 397-1 [3] apply.

4 Mapping Description

The Call Handling capability can be implemented with Parlay/OSA Multi-Party Call Control and Multi-Media Call Control.

It is applicable to ETSI OSA 1.x/2.x/3.x, Parlay/OSA 3.x/4.x/5.x and 3GPP Releases 4/5/6.

5 Sequence Diagrams

5.1 Create Conference

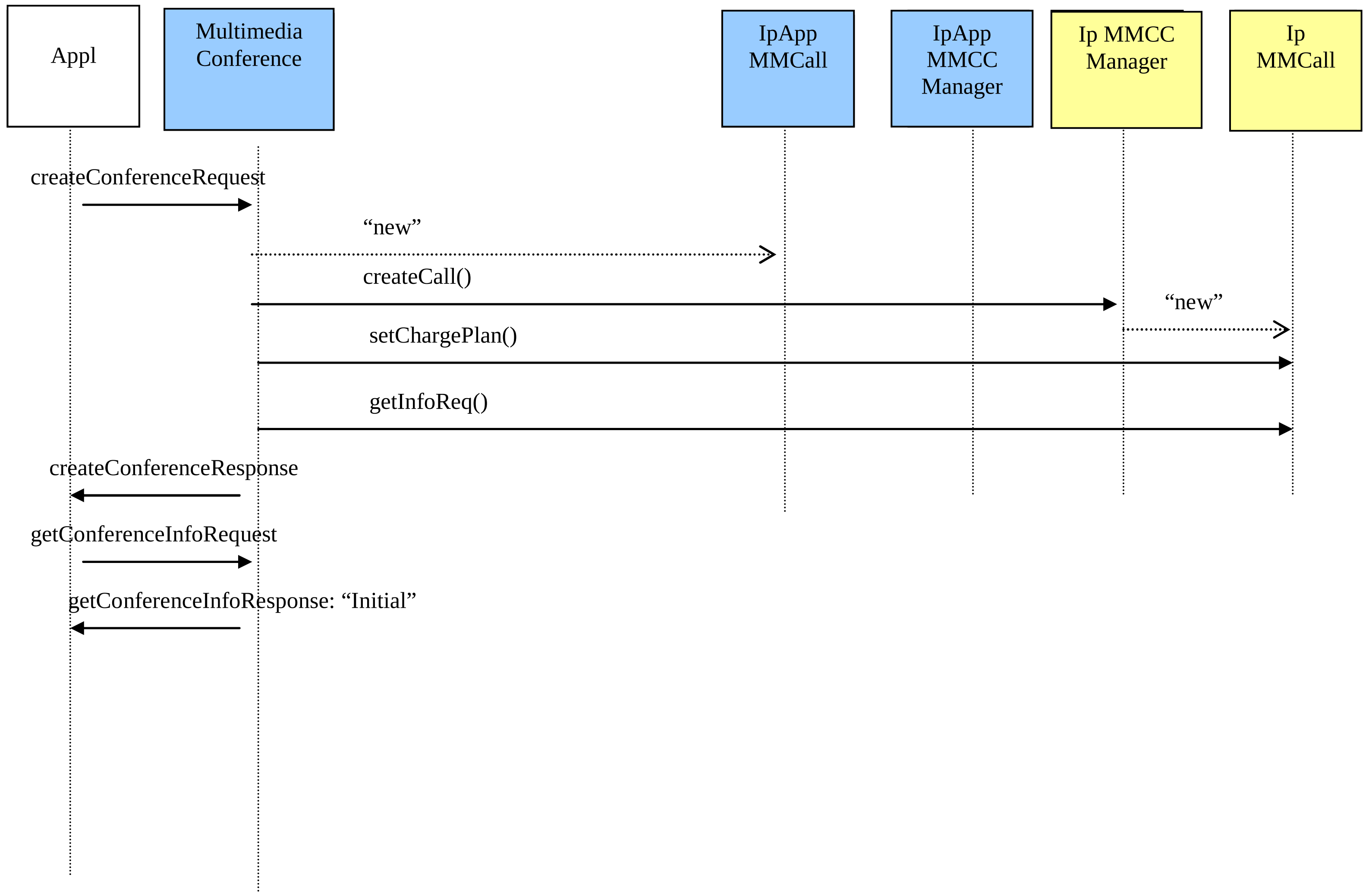

Figure 1

5.2 End Conference

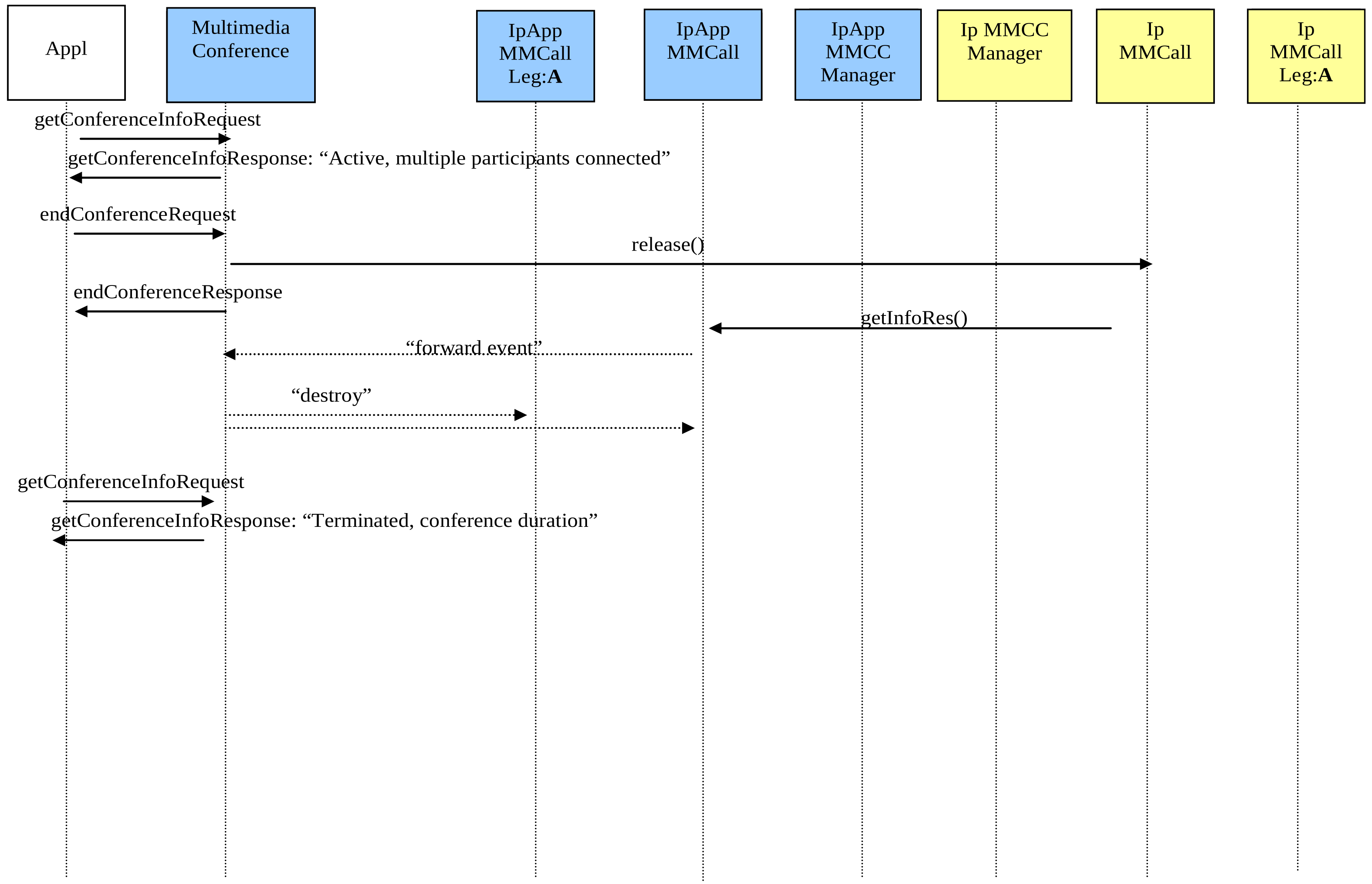

Figure 2

5.3 Invite Participant

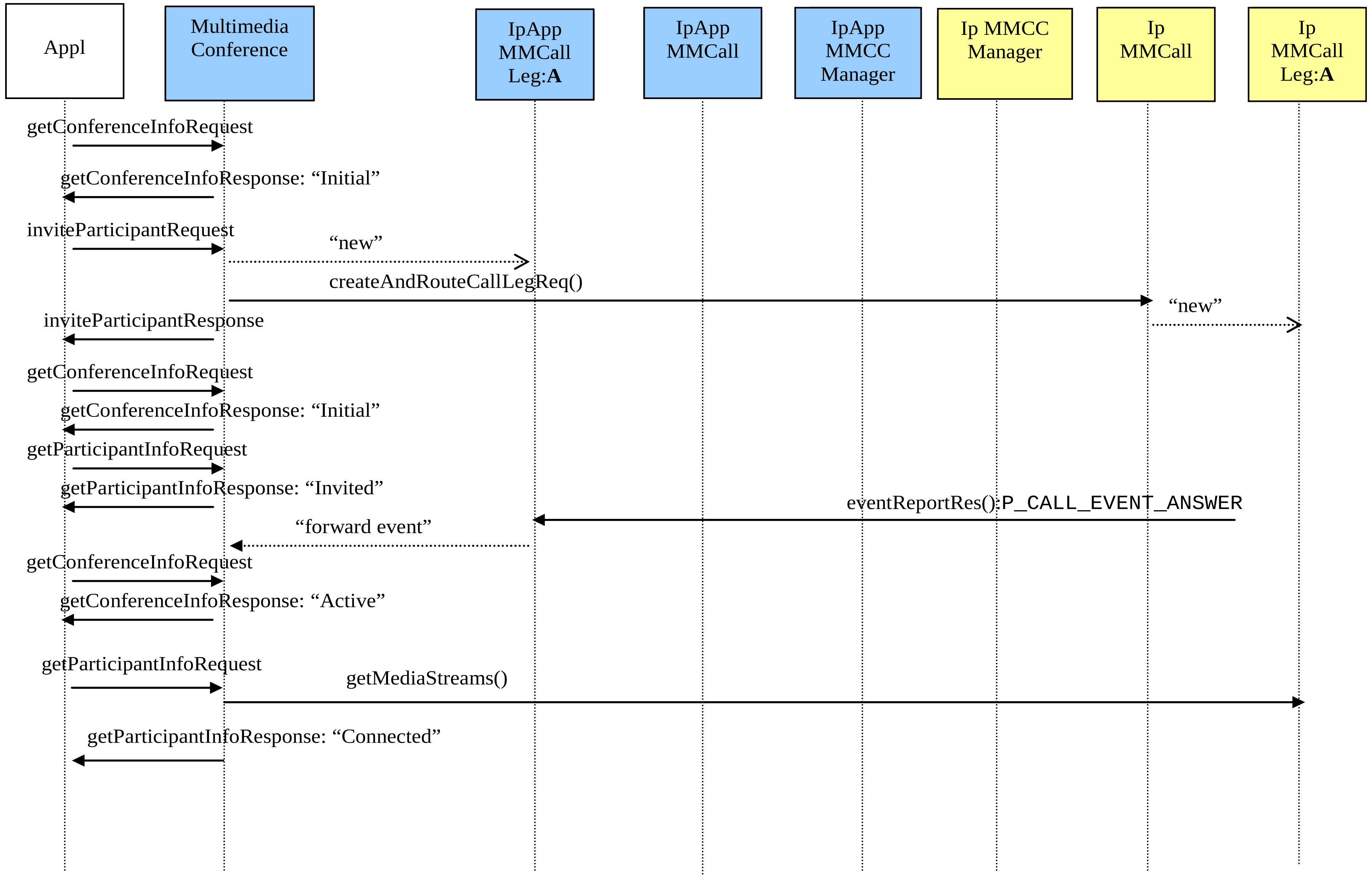
Appl
Multimedia Conference
IpApp MMCall Leg:A
IpApp MMCall
IpApp MMCC Manager
Ip MMCC Manager
Ip MMCall
Ip MMCall Leg:A
getConferenceInfoRequest
getConferenceInfoResponse: “Initial”
inviteParticipantRequest
“new”
createAndRouteCallLegReq()
“new”
inviteParticipantResponse
getConferenceInfoRequest
getConferenceInfoResponse: “Initial”
getParticipantInfoRequest
getParticipantInfoResponse: “Invited”
eventReportRes():P_CALL_EVENT_ANSWER
“forward event”
getConferenceInfoRequest
getConferenceInfoResponse: “Active”
getParticipantInfoRequest
getMediaStreams()
getParticipantInfoResponse: “Connected”

Figure 3

5.4 Invite Participant – Alternative Mapping

There is also another option for mapping **inviteParticipant** operation on Parlay interfaces; this option is presented in the following sequence diagram:

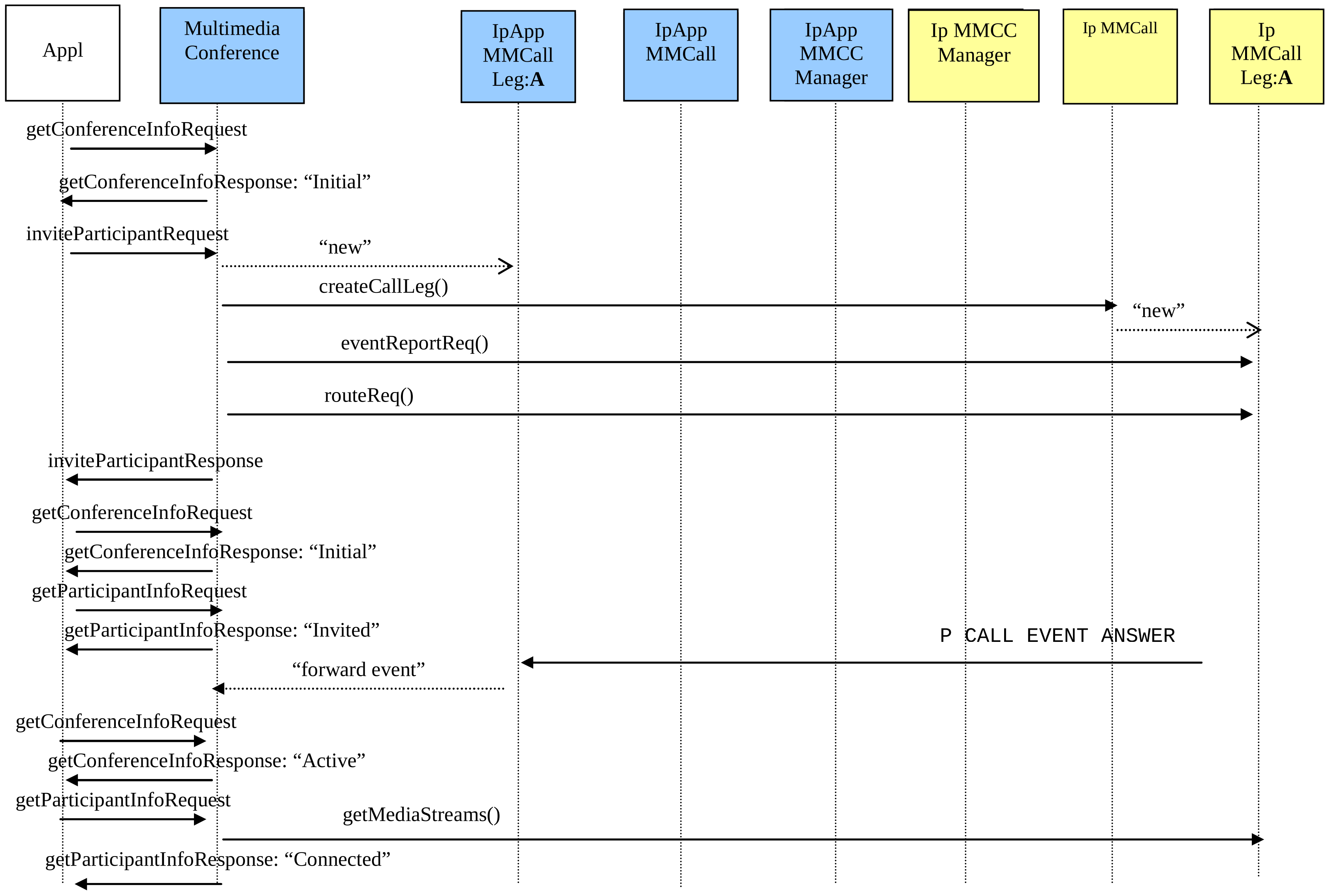

Figure 4

5.5 Disconnect Participant

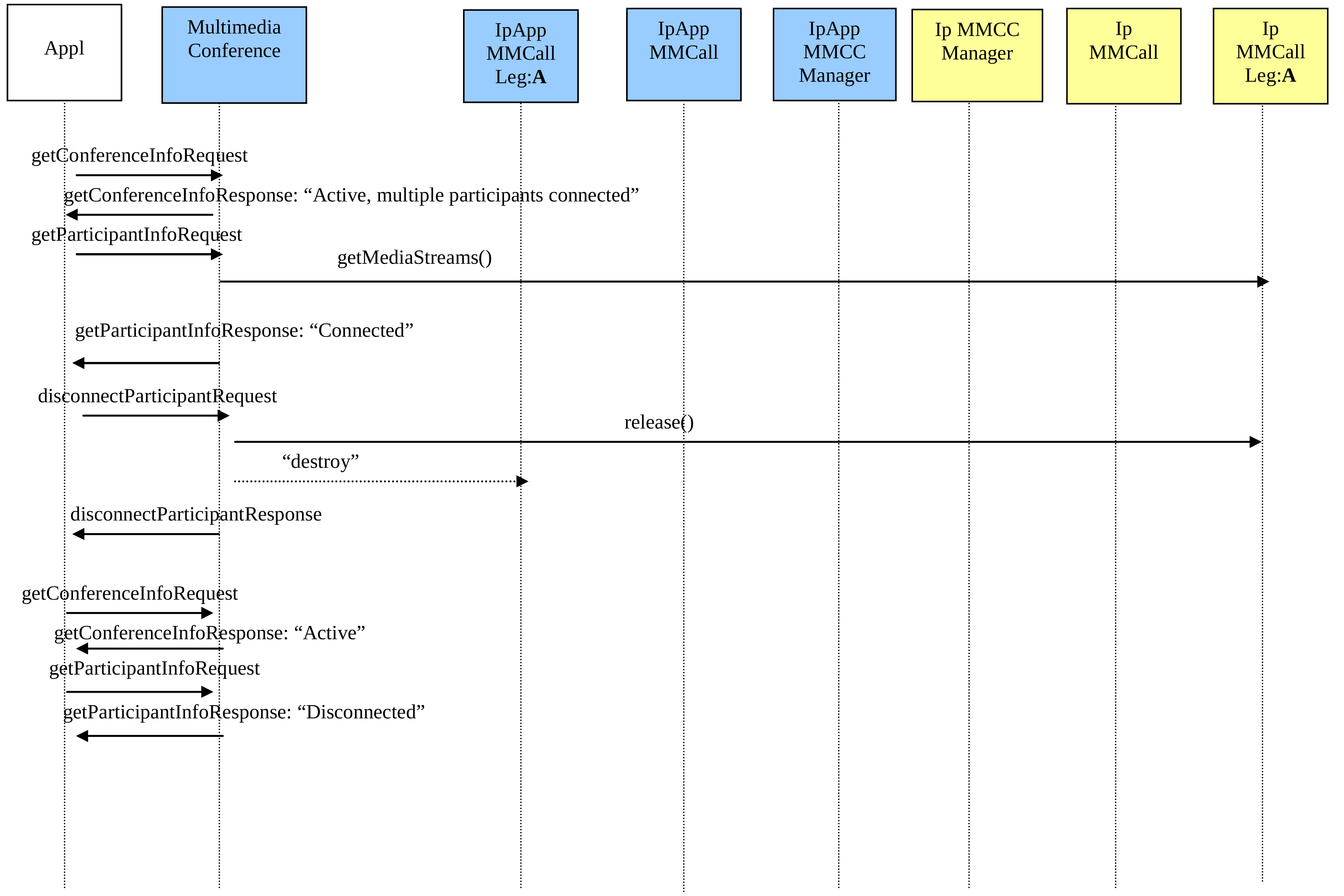
Appl
Multimedia Conference
IpApp MMCall Leg:A
IpApp MMCall
IpApp MMCC Manager
Ip MMCC Manager
Ip MMCall
Ip MMCall Leg:A
getConferenceInfoRequest
getConferenceInfoResponse: "Active, multiple participants connected"
getParticipantInfoRequest
getMediaStreams()
getParticipantInfoResponse: "Connected"
disconnectParticipantRequest
release()
"destroy"
disconnectParticipantResponse
getConferenceInfoRequest
getConferenceInfoResponse: "Active"
getParticipantInfoRequest
getParticipantInfoResponse: "Disconnected"

Figure 5

5.6 Add Media For Participant

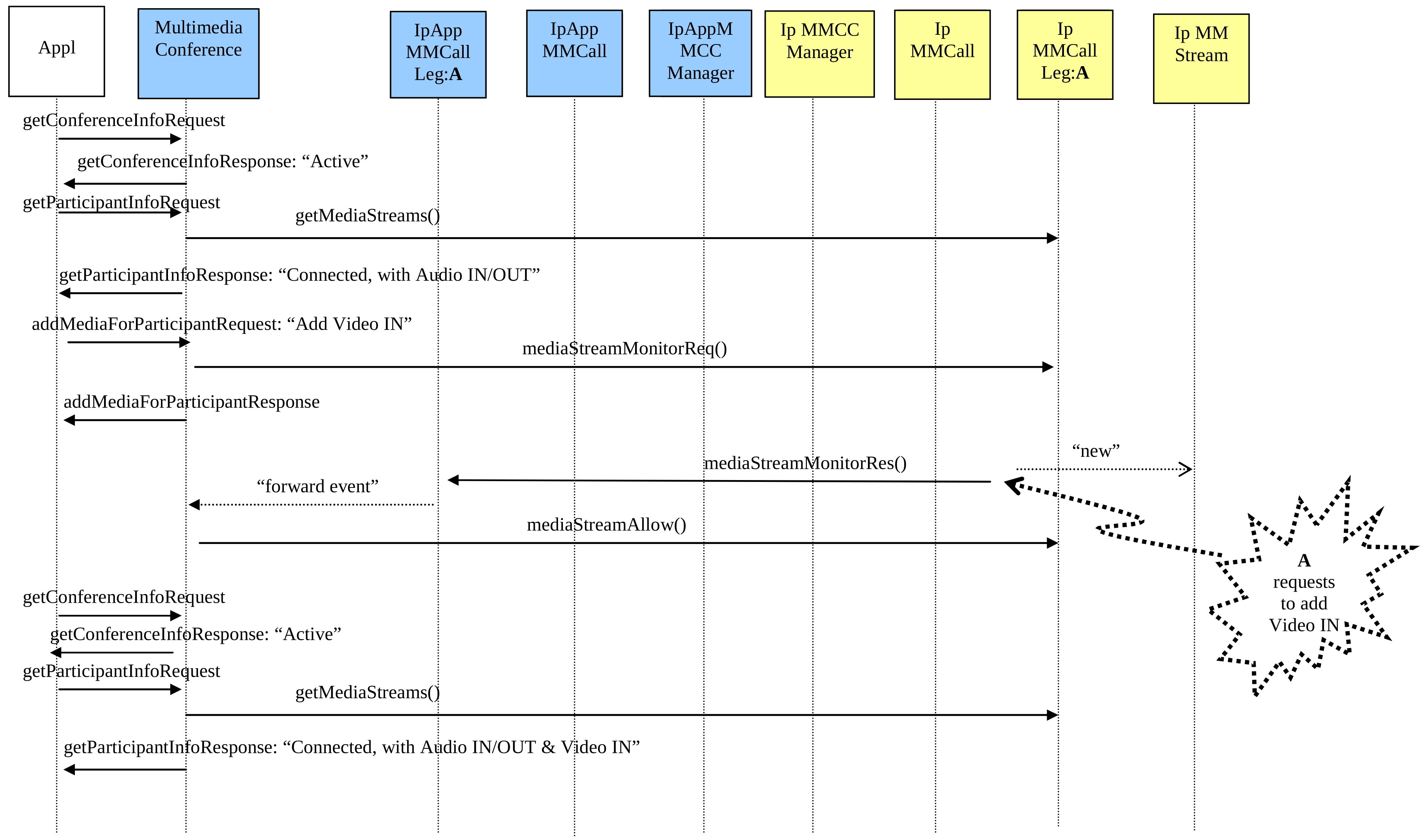
Appl
Multimedia Conference
IpApp MMCall Leg:A
IpApp MMCall
IpAppM MCC Manager
Ip MMCC Manager
Ip MMCall
Ip MMCall Leg:A
Ip MM Stream
getConferenceInfoRequest
getConferenceInfoResponse: "Active"
getParticipantInfoRequest
getMediaStreams()
getParticipantInfoResponse: "Connected, with Audio IN/OUT"
addMediaForParticipantRequest: "Add Video IN"
mediaStreamMonitorReq()
addMediaForParticipantResponse
"new"
mediaStreamMonitorRes()
"forward event"
mediaStreamAllow()
A requests to add Video IN
getConferenceInfoRequest
getConferenceInfoResponse: "Active"
getParticipantInfoRequest
getMediaStreams()
getParticipantInfoResponse: "Connected, with Audio IN/OUT & Video IN"

Figure 6

5.7 Delete Media For Participant

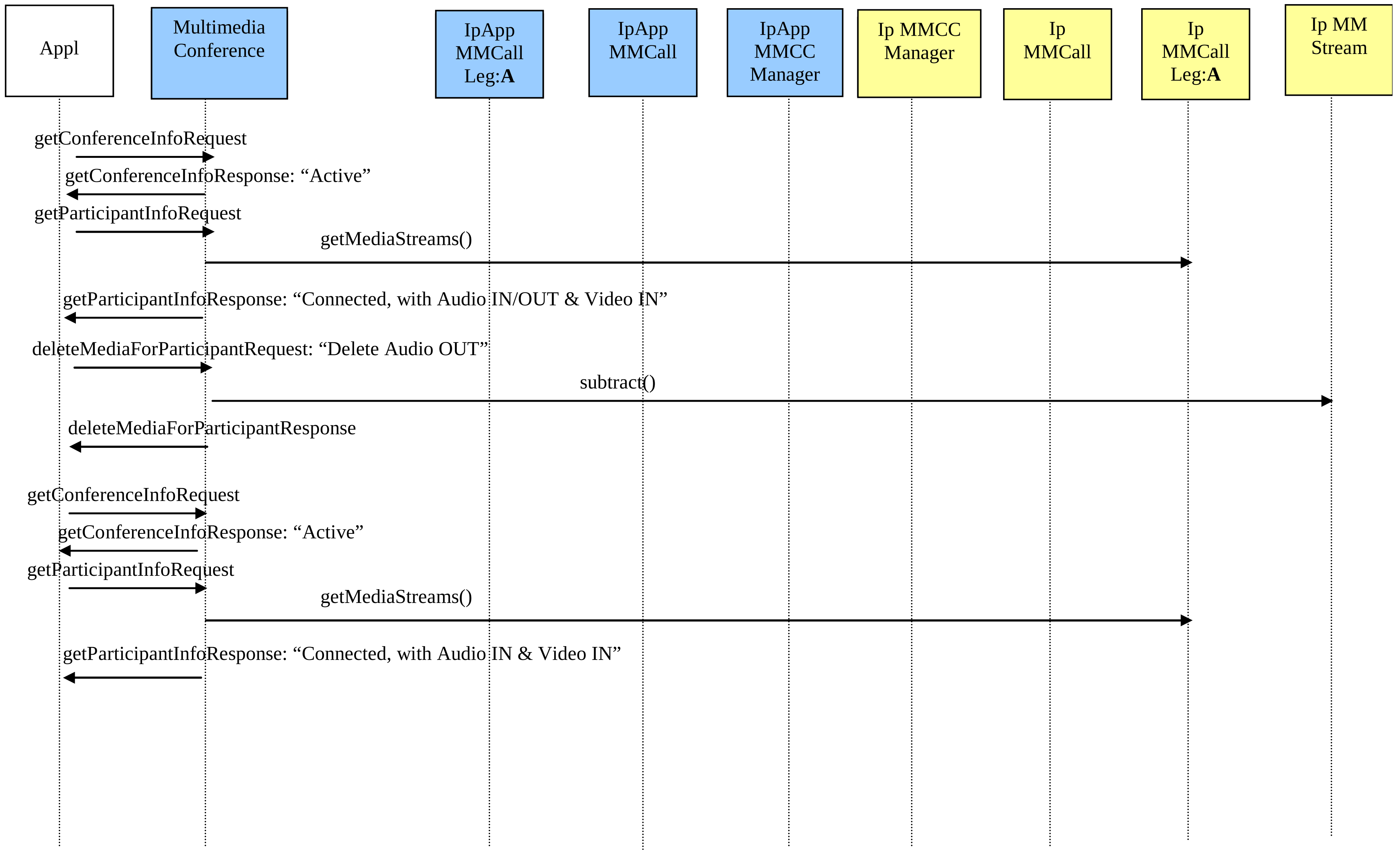
Appl
Multimedia Conference
IpApp MMCall Leg:A
IpApp MMCall
IpApp MMCC Manager
Ip MMCC Manager
Ip MMCall
Ip MMCall Leg:A
Ip MM Stream
getConferenceInfoRequest
getConferenceInfoResponse: "Active"
getParticipantInfoRequest
getMediaStreams()
getParticipantInfoResponse: "Connected, with Audio IN/OUT & Video IN"
deleteMediaForParticipantRequest: "Delete Audio OUT"
subtract()
deleteMediaForParticipantResponse
getConferenceInfoRequest
getConferenceInfoResponse: "Active"
getParticipantInfoRequest
getMediaStreams()
getParticipantInfoResponse: "Connected, with Audio IN & Video IN"

Figure 7

6 Detailed Mapping Information

In this clause will be presented a detailed mapping between the Multimedia Conferencing Parlay X component and the OSA/Parlay specifications. Since in Parlay the Multi Media interfaces inherit from Multi Party interface, related to call control capabilities we can indifferently use both Multi Media or Multi Party interfaces for mapping sequence.

6.1 Operations

6.1.1 createConference

The sequence diagram in clause 5.1 illustrates the flow for the **createConference** operation.

The **createConference** operation is synchronous from the Parlay X client's point of view. It is mapped to following Parlay/OSA methods:

- `IpMultiMediaCallControlManager.createCall`
- `IpMultiMediaCall.setChargePlan`
- `IpMultiMediaCall.getInfoReq`

6.1.1.1 Mapping to `IpMultiMediaCallControlManager.createCall`

The `IpMultiMediaCallControlManager.createCall` operation is invoked with the following parameters:

Name	Type	Comment
appCall	IpAppMultiMediaCallRef	Reference to callback (internal).

The return parameter `TpMultiMediaCallIdentifier` (in particular the field `CallSessionID: TpSessionID`) is mapped to the **conferenceIdentifier (xsd:string)** part returned to the application in the **createConferenceResponse** message.

Parlay exceptions thrown by `IpMultiMediaCallControlManager.createCall` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.1.2 Mapping to `IpMultiMediaCall.setChargePlan`

If the optional **charging** part in the **createConferenceRequest** operation is present (and only in this case), the operation is also mapped to the Parlay request `IpMultiMediaCall.setChargePlan`.

The method `IpMultiMediaCall.setChargePlan` is invoked with the following parameters:

Name	Type	Comment
callSessionID	TpSessionID	Not mapped. [The value provide in the result from `IpMultiMediaCallControlManager.createCall`]
callChargePlan	TpCallChargePlan	Specifies the charge plan to use. It is constructed based on the values provided in the optional **charging** part of **createConferenceRequest**. See the following table for details.

The `callChargePlan` parameter is constructed as follows:

Name	Type	Comment
ChargeOrderType	TpCallChargeOrderCategory	Not mapped
TransparentCharge	TpOctetSet	Specifies an operator-specific charge plan. It is constructed using the value of the **ChargingInformation.code** element provided in the **charging** part.
ChargePlan	TpInt32	Not mapped
AdditionalInfo	TpOctetSet	Descriptive string sent to billing system. It is constructed using the value of the **ChargingInformation.description** element provided in the **charging** part. (May optionally include values of other elements of the **charging** part.)
PartyToCharge	TpCallPartyToChargeType	Not mapped.
PartyToChargeAdditionalInfo	TpCallPartyToChargeAdditional Info	Not mapped

The other parameters of **createConference** operation are not involved in Parlay/OSA mapping operation.

Parlay exceptions thrown by `IpMultiMediaCall.setChargePlan` are mapped to Parlay X exceptions as defined in clause 6.2.

Mapping to `IpMultiMediaCall.getInfoReq`

The method `IpMultiMediaCall.getInfoReq` is invoked with the following parameters:

Name	Type	Comment
callSessionID	TpSessionID	Not mapped. [The value provide in the result from `IpMultiMediaCallControlManager.createCall`]
callInfoRequested	TpCallInfoType	Not mapped. [Set to a value of `P_CALL_INFO_TIMES` to obtain information on relevant call times.]

Parlay exceptions thrown by `IpMultiMediaCall.getInfoReq` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.2 getConferenceInfo

The sequence diagrams in 5.1 through 5.7 illustrates the flow for the **getConferenceInfo** operation.

- The **getConferenceInfo** operation is synchronous from the Parlay X client's point of view. It is mapped indirectly from multiple Parlay/OSA methods which report the connection of the first conference participant, duration of the conference call, the disconnection of the last conference participant or of the conference owner, and the termination of the conference call:`IpAppMultiMediaCallLeg.eventReportRes`
- `IpAppMultiMediaCall.getInfoRes`
- `IpAppMultiMediaCallLeg.callLegEnded`
- `IpAppMultiMediaCall.callEnded`

6.1.2.1 Mapping from `IpAppMultiMediaCallLeg.eventReportRes`

The `IpAppMultiMediaCallLeg.eventReportRes` callback method is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped. [The value provide in the result from `IpMultiMediaCall.create(AndRoute)CallLeg(Req)`]
eventInfo	TpCallEventInfo	Specifies the result of the request to add a participant to the conference. See the following discussion for details of its mapping to the **getConferenceInformation** operation (see note).
NOTE: The `CallMonitorMode` and `AdditionalCallEventInfo` elements of `eventInfo` are not mapped.		

If the **getConferenceInformationRequest** message is received from the application before `eventReportRes` is *successfully* invoked for the first conference participant (where *successfully* means the `CallEventType` element of the `eventInfo` parameter has a value of `P_CALL_EVENT_ANSWER)`, then the **conferenceInformation** part of the **getconferenceInformationResponse** message is constructed as follows:

- ConferenceInfo.status has a value of Initial
- ConferenceInfo.startTime has the following value: the date/time that the `IpMultiMediaCallControlManager.createCall` operation was successfully invoked
- ConferenceInfo.duration is not applicable
- The other elements of the ConferenceInfo structure are not mapped from `eventInfo`.

If the **getConferenceInformationRequest** message is received from the application after `eventReportRes` is *successfully* invoked for the first conference participant, and before the conference terminates, then the **conferenceInformation** part of the **getconferenceInformationResponse** message is constructed as follows:

- ConferenceInfo.status has a value of Active
- ConferenceInfo.startTime has the following value: the date/time that the `IpMultiMediaCallControlManager.createCall` operation was successfully invoked
- ConferenceInfo.duration has a value derived by subtracting the value of the `CallEventTime` element of the `eventInfo` parameter from the current date/time.
- The other elements of the ConferenceInfo structure are not mapped from `callInfoReport`.

6.1.2.2 Mapping from `IpAppMultiMediaCall.getInfoRes`

When call information is available, the `IpAppMultiMediaCall.getInfoRes` callback is invoked. It is expected to contain call info times relevant information related to callSessionID indicates in the request (corresponding to **conferenceIdentifier** for Multimedia Conferencing).

The method `IpAppMultiMediaCall.getInfoRes` is invoked with the following parameters:

Name	Type	Comment
callSessionID	TpSessionID	Not mapped. [The value provide in the result from `IpMultiMediaCall.createCall`]
callInfoReport	TpCallInfoReport	Specifies the call information requested. See the following discussion for details of its mapping to the **getConferenceInformation** operation (see note).
NOTE: The `CallInfoType`, `CallConnectedToResourceTime` and `Cause` elements of `callInfoReport` are not mapped.		

If the **getConferenceInformationRequest** message is received from the application after invocation of the `IpAppMultiMediaCall.getInfoRes` method, then the **conferenceInformation** part of the **getconferenceInformationResponse** message is constructed as follows:

- ConferenceInfo.status has a value of Terminated
- ConferenceInfo.startTime has the value provided in the `CallInitiationStartTime` element of the `callInfoReport` parameter.
- ConferenceInfo.duration has a value derived by subtracting the values of the `CallEndTime` and `CallConnectedToDestinationTime` elements of the `callInfoReport` parameter.
- The other elements of the ConferenceInfo structure are not mapped from `callInfoReport`.

6.1.2.3 Mapping from `IpAppMultiMediaCallLeg.callLegEnded`

If the **getConferenceInformationRequest** message is received from the application after invocation of the `IpAppMultiMediaCallLeg.callLegEnded` method, and the termination is for either the last participant in the conference call or the conference call owner, then the **conferenceInformation** part of the **getconferenceInformationResponse** message is constructed as follows:

- ConferenceInfo.status has a value of Terminated
- ConferenceInfo.startTime. If the `IpAppMultiMediaCall.getInfoRes` method has already been invoked, then the value is derived from the `CallInitiationStartTime` element of the `callInfoReport` parameter of the `IpAppMultiMediaCall.getCallInfoRes` method. Otherwise the value is the date/time that the `IpMultiMediaCallControlManager.createCall` operation was successfully invoked.
- ConferenceInfo.duration. If the `IpAppMultiPartyCall.getInfoRes` method has already been invoked, then the value is derived by subtracting the values of the `CallEndTime` and `CallConnectedToDestinationTime` elements of the `callInfoReport` parameter of the `IpAppMultiPartyCall.getInfoRes` method. Otherwise, the value is derived by subtracting the value of the date/time when the conference entered the "Active" state (i.e. the first participant successfully connected) from the date/time that this `IpAppMultiMediaCallLeg.callLegEnded` method was invoked.

NOTE: For the case where the `callLegEnded` invocation is for the conference call owner, and there are other participants on the conference, then the Multimedia Conference web service terminates the conference call by invoking the `IpMultiMediaCall.release` method.

6.1.2.4 Mapping from `IpAppMultiMediaCall.callEnded`

If the **getConferenceInformationRequest** message is received from the application following the termination of the call, then the **conferenceInformation** part of the **getconferenceInformationResponse** message is constructed as follows:

- ConferenceInfo.status has a value of Terminated
- ConferenceInfo.startTime. If the `IpAppMultiMediaCall.getInfoRes` method has already been invoked, then the value is derived from the `CallInitiationStartTime` element of the `callInfoReport` parameter of the `IpAppMultiMediaCall.getCallInfoRes` method. Otherwise the value is the date/time that the `IpMultiMediaCallControlManager.createCall` operation was successfully invoked.
- ConferenceInfo.duration. If the `IpAppMultiPartyCall.getInfoRes` method has already been invoked, then the value is derived by subtracting the values of the `CallEndTime` and `CallConnectedToDestinationTime` elements of the `callInfoReport` parameter of the `IpAppMultiPartyCall.getInfoRes` method. Otherwise, the value is derived by subtracting the value of the date/time when the conference entered the "Active" state (i.e. the first participant successfully connected) from the date/time that this `IpAppMultiMediaCall.callEnded` method was invoked.

6.1.3 endConference

The sequence diagram in clause 5.2 illustrates the flow for the **endConference** operation.

The **endConference** operation is synchronous from the Parlay X client's point of view. It is mapped to the Parlay request `IpMultiMediaCall.release.`

6.1.3.1 Mapping to `IpMultiMediaCall.release`

The method `IpMultiMediaCall.release` is invoked with the following parameters:

Name	Type	Comment
callSessionID	TpSessionID	Reference to call identifier (corresponding to **conferenceIdentifier**) related to call to be released. This parameter is mapped on **conferenceIdentifier**.
cause	TpReleaseCause	Not mapped. Assigned a value indicating application-initiated call termination

Parlay exceptions thrown by `IpMultiMediaCall.release` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.4 inviteParticipant

The sequence diagrams in 5.3 and 5.4 illustrate the flow for the **inviteParticipant** operation.

The **inviteParticipant** operation is synchronous from the Parlay X client's point of view. It maps to/from the following Parlay/OSA methods:

- `IpMultiMediaCall.createAndRouteCallLegReq`, as illustrated in 5.3, `OR`

  `{IpMultiMediaCall.createCallLeg, IpMultiMediaCallLeg.eventReportReq, IpMultiMediaCallLeg.routeReq}`, as illustrated in 5.4

6.1.4.1 Mapping to `IpMultiMediaCall.createAndRouteCallLegReq`

The method `IpMultiMediaCall.createAndRouteCallLegReq` is invoked with the following parameters:

Name	Type	Comment
callSessionID	TpSessionID	Not mapped. [The value provide in the result from `IpMultiMediaCallControlManager.createCall`]
eventRequested	TpCallEventRequestSet	Not mapped. [Requests call-related event reports: i.e. including at least the "Answer" event. The `MonitorMode` element of each requested event report should have a value of `P_CALL_MONITOR_MODE_NOTIFY`]
targetAddress	TpAddress	Specifies the destination party to which the call should be routed. It is constructed based on the URI provided in the **participant** part of **inviteParticipantRequest**, mapped as described in TR 102 397-1 [3].
originatingAddress	TpAddress	Not mapped.
appInfo	TpCallAppInfoSet	Not mapped.
appLegInterface	IpAppMultiMediaCallLegRef	Not mapped. [Specifies a reference to the application interface that implements the callback interface for the new call leg. Requested events will be reported by the `eventReportRes()` method on this interface.]

The result from `IpMultiMediaCall.createAndRouteCallLegReq` is of type `TpMultiMediaCallLegIdentifier` and is used internally to correlate the callbacks. It is not mapped to the Parlay X interface.

Parlay exceptions thrown by `IpMultiMediaCall.createAndRouteCallLegReq` are mapped to Parlay X exceptions as defined in clause 6.2.

An alternative to mapping to the `IpMultiMediaCall.createAndRouteCallLegReq` convenience method is a mapping to the following discrete method invocations

- `IpMultiMediaCall.createCallLeg`
- `IpMultiMediaCallLeg.eventReportReq`
- `IpMultiMediaCallLeg.routeReq`

6.1.4.1.1 Alternative Mapping to IpMultiMediaCall.createCallLeg

The method IpMultiMediaCall.createCallLeg is invoked with the following parameters:

Name	Type	Comment
callSessionID	TpSessionID	Not mapped: the result from the invocation of IpMultiMediaCallControlManager.createCall
appCallLeg	IpAppMultiMediaCallLegRef	Not mapped. [Specifies a reference to the application interface that implements the callback interface for the new call leg. Requested events will be reported by the `eventReportRes()` operation on this interface.]

The result from IpMultiMediaCall.createCallLeg is of type TpMultiMediaCallLegIdentifier and is not mapped to the Parlay X interface.

Parlay exceptions thrown by IpMultiMediaCall.createCallLeg are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.4.1.2 Alternative Mapping to `IpMultiMediaCallLeg.eventReportReq`

The method IpMultiMediaCallLeg.eventReportReq is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of IpMultiMediaCall.createCallLeg
eventRequested	TpCallEventRequestSet	Not mapped. [Requests call-related event reports: i.e. including at least the "Answer" event. The `MonitorMode` element of each requested event report should have a value of `P_CALL_MONITOR_MODE_NOTIFY`]

Parlay exceptions thrown by IpMultiMediaCallLeg.eventReportReq are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.4.1.3 Alternative Mapping to IpMultiMediaCallLeg.routeReq

The method IpMultiMediaCallLeg.routeReq is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of IpMultiMediaCall.createCallLeg
targetAddress	TpAddress	Specifies the destination party to which the call should be routed. It is constructed based on the URI provided in the **participant** part of **inviteParticipantRequest**, mapped as described in TR 102 397-1 [3]..
originatingAddress	TpAddress	Not mapped.
appInfo	TpCallAppInfoSet	Not mapped.
connectionProperties	TpCallLegConnection Properties	Not mapped. Specifies the properties of the connection: • `AttachMechanism` = `P_CALLLEG_ATTACH_IMPLICITLY`, i.e. the call leg should be attached implicitly to the call

Parlay exceptions thrown by IpMultiMediaCallLeg.routeReq are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.5 disconnectParticipant

The sequence diagram in clause 5.5 illustrates the flow for the **disconnectParticipant** operation.

The **disconnectParticipant** operation is synchronous from the Parlay X client's point of view. It is mapped to the Parlay request `IpMultiMediaCallLeg.release.`

6.1.5.1 Mapping to IpMultiMediaCallLeg.release

The method IpMultiMediaCallLeg.release is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of IpMultiMediaCall.create(AndRoute)CallLeg(Req)
cause	TpReleaseCause	Not mapped. Assigned a value indicating application-initiated call termination

Parlay exceptions thrown by IpMultiMediaCallLeg.release are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.6 addMediaForParticipant

The sequence diagram in clause 5.6 illustrates the flow for the **addMediaForParticipant** operation.

The **addMediaForParticipant** operation is synchronous from the Parlay X client's point of view. This mapping operation allows the addition of a new media stream which was previously requested by a participant on their own terminal equipment. It does not allow the application to directly add a new media stream since this capability is not supported using Parlay requests. For the latter kind of operation we need a mapping to other protocols or platforms (e.g. an invite SIP) which is not presented in the present document.

This operation is mapped to the Parlay request IpMultiMediaCallLeg.mediaStreamAllow, after invoking the IpMultiMediaCallLeg.mediaStreamMonitorReq method and the corresponding asynchronous callback operation IpAppMultiMediaCallLeg.mediaStreamMonitorRes. In this way, when a participant decides to add a stream on their own terminal, the monitor result is sent to the Multimedia Conference web service (by IpAppMultiMediaCallLeg.mediaStreamMonitorRes callback invocation); depending on the **media** and **mediaDirection** parameters in **addMediaForParticipant**, the web service invokes the IpMultiMediaCallLeg.mediaStreamAllow method in order to allow or deny the permission for adding the new media stream on the participant's terminal.

The **addMediaForParticipant** operation maps to/from the following Parlay/OSA methods:

- IpMultiMediaCallLeg.mediaStreamMonitorReq
- IpAppMultiMediaCallLeg.mediaStreamMonitorRes
- IpMultiMediaCallLeg.mediaStreamAllow

6.1.6.1 Mapping to IpMultiMediaCallLeg.mediaStreamMonitorReq

The method IpMultiMediaCallLeg.mediaStreamMonitorReq is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of IpMultiMediaCall.create(AndRoute)CallLeg(Req)
mediaStreamEventCriteria	TpMediaStream RequestSet	Consists of a single set element defining the media stream for which to monitor. It is constructed based on the values provided in the **media** and **mediaDirection** parts of the **addMediaForParticipantRequest** message. See the following table for details.

The `mediaStreamEventCriteria` parameter is constructed as follows:

Name	Type	Comment
Direction	TpMediaStreamDirection	**MediaDirection** values map to `Direction` values as follows: **In** maps to `P_RECEIVE_ONLY` **Out** maps to `P_SEND_ONLY` **InOut** maps to `P_SEND_RECEIVE`
DataTypeRequest	TpMediaStreamDataType Request	**Media** values map to `DataTypeRequest` values as follows: **Audio** maps to `P_AUDIO_CAPABILITIES`. There is no mapping to the associated `Audio` element, which identifies the audio codec capabilities. This element value is defined by policy. **Video** maps to `P_VIDEO_CAPABILITIES`. There is no mapping to the associated `Video` element, which identifies the video codec capabilities. This element value is defined by policy. **Chat** maps to `P_DATA_ CAPABILITIES`. There is no mapping to the associated `Data` element, which defines the lower threshold for the maxBitRate parameter. This element value is defined by policy. **Data** maps to `P_DATA_CAPABILITIES`. There is no mapping to the associated `Data` element, which defines the lower threshold for the maxBitRate parameter. This element value is defined by policy.
MediaMonitorMode	TpCallMonitorMode	Not mapped. Set to a value of `P_CALL_MONITOR_MODE_INTERRUPT` if the web service is required by policy to explicitly authorize addition of the multimedia stream (i.e. by invoking the `IpMultiMediaCallLeg.mediaStreamAllow` method). Otherwise set to a value of `P_CALL_MONITOR_MODE_NOTIFY.`
EventType	TpMediaStreamEvent Type	Not mapped. Set to a value of `P_MEDIA_STREAM_ADDED`

Parlay exceptions thrown by `IpMultiMediaCallLeg.mediaStreamMonitorReq` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.6.2 Mapping from `IpAppMultiMediaCallLeg.mediaStreamMonitorRes`

The method `IpAppMultiMediaCallLeg.mediaStreamMonitorRes` is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of `IpMultiMediaCall.create(AndRoute)CallLeg(Req)`
streams	TpMediaStreamSet	Specifies all the media streams that are added. Note that this can be more than one media stream, if multiple invocations of `IpMultiMediaCallLeg. mediaStreamMonitorReq` have occurred for this participant. For each set element the validation of the `TpMediaStream` data type is specified in the following table. Validated streams are "allowed" by the web service by invoking the `mediaStreamMonitorReq` method (ref 0).
type	TpMediaStreamEventType	Not mapped. Set to a value of `P_MEDIA_STREAM_ADDED`

Each element of the `streams` set is of type `TpMediaStream` and is validated against the **media** and **mediaDirection** parts of a previously received **addMediaForParticipantRequest** message, as follows:

Name	Type	Comment
Direction	TpMediaStreamDirection	`Direction` matches with **MediaDirection** as follows: `P_RECEIVE_ONLY` matches with **In** or **InOut** `P_SEND_ONLY` matches with **Out** or **InOut** `P_SEND_RECEIVE` matches with **InOut** only (or with two previous **addMediaForParticipantRequest** messages for this participant and the same `DataType`, with **MediaDirection** values of **In** and **Out**.)
DataType	TpMediaStreamDataType	Defines the type of the reported media stream. It is identical to `TpMediaStreamDataTypeRequest`, only now the values are not used as a mask, but the actual codec should be indicated for audio and video. For data the actual maximum bit rate is indicated. `DataType` values match with **Media** values as follows: `P_AUDIO_CAPABILITIES` matches **Audio**. [The associated `Audio` element, which identifies specific audio codec capabilities, can only be validated against defined web service policy, if any.] `P_VIDEO_CAPABILITIES` matches **Video**. [The associated `Video` element, which identifies specific video codec capabilities, can only be validated against defined web service policy, if any.] `P_DATA_CAPABILITIES` matches **Data**. [The associated `Data` element, which identifies the lower threshold for the maxBitRate parameter, can only be validated against defined web service policy, if any.]
ChannelSessionID	TpSessionID	Not mapped. Identifies this specific media stream in the current set associated with the call leg (participant) [This parameter is used to reference the media stream when invoking the `IpMultiMediaCallLeg.mediaStreamAllow` method.]
MediaStream	IpMultiMediaStream	Not mapped. A reference to the interface for this bi-directional information stream that is associated with the call leg (participant)

6.1.6.3 Mapping to `IpMultiMediaCallLeg.mediaStreamAllow`

The method `IpMultiMediaCallLeg.mediaStreamAllow` is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of `IpMultiMediaCall.create(AndRoute)CallLeg(Req)`
mediaStreamList	TpSessionIDSet	Refers to the media stream(s) (the set of `ChannelSessionID` fields) received in the `streams` parameter of the `IpAppMultiMediaCallLeg.mediaStreamMonitorRes()` method, which have been validated: i.e. they match the values provided in the **media** and **mediaDirection** parts of previously received **addMediaForParticipantRequest** message(s), as described in 6.1.6.2.

Parlay exceptions thrown by `IpMultiMediaCallLeg.mediaStreamAllow` are ignored and not mapped to Parlay X exceptions.

6.1.7 deleteMediaForParticipant

The sequence diagram in clause 5.7 illustrates the flow for the **deleteMediaForParticipant** operation.

The **deleteMediaForParticipant** operation is synchronous from the Parlay X client's point of view. The meaning for this mapping operation is only to subtract a media stream for a participant and not to change media stream characteristics (e.g. media direction).

The **deleteMediaForParticipant** operation maps to the following Parlay/OSA methods:

- `IpMultiMediaCallLeg.getMediaStreams`, (if necessary in order to obtain a reference to the appropriate instance of the `IpMultiMediaStream` interface)
- `IpMultiMediaStream.subtract.`

6.1.7.1 Mapping to `IpMultiMediaCallLeg.getMediaStreams`

The method `IpMultiMediaCallLeg.getMediaStreams` is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of `IpMultiMediaCall.create(AndRoute)CallLeg(Req)`

The result from `IpMultiMediaCallLeg.getMediaStreams` is of type `TpMediaStreamSet.` It defines one or more media streams presently assigned to this conference call participant. Each media stream is of type `TpMediaStream` and is matched against the **media** and **mediaDirection** parts of the **deleteMediaForParticipantRequest** message, as described in 6.1.6.2.

- If there is at least one match, then the Multimedia Conference web service uses the associated {ChannelSessionID, MediaStream} element pair(s) to invoke the `IpMultiMediaStream.subtract` method
- If there is no match, then the web service raises a Parlay X exception (**SVC0002**)

Parlay exceptions thrown by `IpMultiMediaCallLeg.getMediaStreams` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.7.2 Mapping to `IpMultiMediaStream.subtract`

The method `IpMultiMediaStream.subtract` is invoked with the following parameters:

Name	Type	Comment
mediaStreamSessionID	TpSessionID	Reference to sessionID for the media stream to be subtracted. (One of) the `ChannelSessionID` field(s) extracted from either of the following sources:the `streams` field of the `IpAppMultiMediaCallLeg. mediaStreamMonitorRes()` method the return parameter of the `IpMultiMediaCallLeg. getMediaStreams()` method •

Parlay exceptions thrown by `IpMultiMediaStream.subtract` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.8 getParticipantInfo

This operation, illustrated in 5.3 through 5.7, retrieves information described in the **ParticipantInfo** complex data type. This information may be managed internally by the Multimedia Conference web service. While the participant's call leg is active (i.e. **ParticipantInfo.status** = **Connected**) detailed media stream information for the participant may also be retrieved directly from the network by invoking the `IpMultiMediaCallLeg.getMediaStreams` method.

The **getParticipantInfo** operation is synchronous from the Parlay X client's point of view. It maps to the following Parlay/OSA methods:

- `IpMultiMediaCallLeg.getMediaStreams`

6.1.8.1 Mapping to `IpMultiMediaCallLeg.getMediaStreams`

The method `IpMultiMediaCallLeg.getMediaStreams` is invoked with the following parameters:

Name	Type	Comment
callLegSessionID	TpSessionID	Not mapped: the result returned from the invocation of `IpMultiMediaCall.create(AndRoute)CallLeg(Req)`

The result from `IpMultiMediaCallLeg.getMediaStreams` is of type `TpMediaStreamSet.` It defines one or more media streams presently assigned to this conference call participant. The content of the media stram(s) is mapped to the **codecAudio/Video/In/Out** element(s) of the **ParticipantInfo** complex data type as follows:

- All streams with a `DataType` value of `P_DATA_CAPABILITIES` are ignored
- All streams with a `DataType` value of `P_AUDIO_CAPABILITIES` and a `Direction` value of either `P_RECEIVE_ONLY` or `P_SEND_RECEIVE` are mapped to the **codecAudioIn** element: in the case of multiple streams, the values of the `Audio` element(s) may be expressed in the form of a concatenated string: e.g. 'P_UMTS_AMR_NB, P_G722_48K'
- All streams with a `DataType` value of `P_AUDIO_CAPABILITIES` and a `Direction` value of either `P_SEND_ONLY` or `P_SEND_RECEIVE` are mapped to the **codecAudioOut** element: in the case of multiple streams, the values of the `Audio` element(s) may be expressed in the form of a concatenated string
- All streams with a `DataType` value of `P_VIDEO_CAPABILITIES` and a `Direction` value of either `P_RECEIVE_ONLY` or `P_SEND_RECEIVE` are mapped to the **codecVideoIn** element: in the case of multiple streams, the values of the `Video` element(s) may be expressed in the form of a concatenated string
- All streams with a `DataType` value of `P_VIDEO_CAPABILITIES` and a `Direction` value of either `P_SEND_ONLY` or `P_SEND_RECEIVE` are mapped to the **codecVideoOut** element: in the case of multiple streams, the values of the `Video` element(s) may be expressed in the form of a concatenated string

Parlay exceptions thrown by `IpMultiMediaCallLeg.getMediaStreams` are mapped to Parlay X exceptions as defined in clause 6.2.

6.1.9 getParticipants

This operation requests a report of the current status of each participant of a specified multimedia conference. Similar to the **getParticipantInfo** operation, it is not directly mapped to Parlay/OSA methods.

6.2 Exceptions

6.2.1 Mapping from `TpCallError`

The following table indicates how `TpCallError` value are mapped to Parlay X exceptions:

Value	Exception	Notes
P_CALL_ERROR_UNDEFINED	SVC0001	
P_CALL_ERROR_INVALID_ADDRESS	SVC0004	
P_CALL_ERROR_INVALID_STATE	SVC0001	
P_CALL_ERROR_RESOURCE_UNAVAILABLE	SVC0001	

6.2.2 Mapping from Parlay/OSA Method Exceptions

For the present document, the mapping of Parlay/OSA API method exceptions to Parlay X Web Service exceptions is common and defined in TR 102 397-1 [3]. There are no service-specific exception mappings.

7 Additional Notes

No additional notes.

History

Document history		
V1.1.1	August 2005	Publication