
Application Programming Interface Document

Media Sever API Specification

Version 1.0

Document Information**Document Sign Off**

Project Manager (Solution Architecture & QA)	Mr. INAMULLAH
Development Lead (Media Server Project)	Mr. Waqqas Jabbar
Documentation Team	Technical Writing Department

Document Information

Version #	1.0
Revision Date	April 22, 2008.
Prepared By	Shafaq Irshad.

History**Document Version Control**

Date	Revision	Author	Description
April 22, 2008		Shafaq Irshad	The document contains information of various Media Server APIs

Document Purpose:

The document is designed to give comprehensive and quick information about Media Server APIs. It further contains detailed structure explanation of Media Server APIs.

Contents

2. Initialization & Configuration APIs	7
2.1.1. InitMediaServer	7
3. SIP Call Back APIs	8
3.1.1. HandleSIPRequest	8
3.1.2. HandleSIPResponse	8
4. Conference Processing APIs	9
4.1.1. HandleConferenceRequest	9
4.1.2. HandleConferenceResponse	9
4.1.3. CreateConference	10
4.1.4. RemoveConference	10
4.1.5. AddParticipant	11
4.1.6. RemoveParticipant	11
4. Non Conference Processing API	12
4.2.1. HandleNonConferenceRequest	12
4.2.2. HandleNonConferenceResponse	12
6. User Call Leg API	15
6.1.1. EventProcessor	15
6.1.2. SendInfo	15
9. MSCML Processing API	18
9.1.1. EventProcessor	18
9.1.2. StartTimer	18
9.1.3. StopTimer	19
9.1.4. SendMSCMLResponse	19
9.1.5. Play	19
9.1.6. Collect	20
9.1.7. Record	20
9.1.8. Stop	21
10. Media Processing API	23

10.1.1. AllocateMedia	23
10.1.2. ConfigureMedia	23
10.1.3. UnConfigureMedia	24
10.1.4. UnallocateMedia	24

1) MsRetCode

```
Typedef enum MsRetCode  
{  
    MSE_Success,  
    MSE_InvalidValue  
    MSE_InvalidID  
    MSE_NotFound  
    MSE_AlreadyExists  
    MSE_InsufficientResources,  
  
    } MsRetCode_t
```

2) MSInitData

```
MsInitData  
{  
    String                strNonConfUri  
    SDP*                 pLocalCap  
    SIP_Config*         pSipConfig  
    Framework_Config*   pFrameworkConfig  
  
    } MsInitData_t;
```

Structure Explanation

Fields:

StrNonConfigUri

URI for non – conference processing.

pLocalCap

Handle of Local Media Capabilities.

pSipConfig

Pointer to SIP Stack Configuration.

pFrameworkCongi

Pointer to Framwork Configurations.

2.1 Initialization & Configuration APIs

2.1.1. InitMediaServer

MsRetCode *InitMediaServer*(*MsInitData** *pInitData*);

Purpose

This function initializes the Media Server.

Parameters

Name	Type	In/out	Description
pInitData	MsInitData*	In	Handle to initialization data of Media Server.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

3) Conference Structure

```

MsConference
{
    String                StrConfId;
    MsCallLeg*           pCtrlLeg;
    MsCallLegList*      pParticipantList;
    unsigned int         uiCurrNumPatipant;
    unsigned int         uiMaxAllowedPartcipant;
} MsConference_t;

```

Structure Explanation

Fields:

StrConfId

This string is a conference identifier of this conference.

pCtrlLeg

Control Leg associated with this Conference.

pParticipantList

List of participants in the conference.

uiCurrNumPartipant

This field identifies current number of participants in the conference.

uiMaxAllowedParticipant

This field identifies maximum allowed participants in the conference.

3.1 SIP Call Back APIs

3.1.1. HandleSIPRequest

MsRetCode HandleSIPRequest (SipCallLeg pCallLeg ,SipMsg* pSipMsg);*

Purpose

The purpose of this API is to handle SIP requests received by Media Server.

Parameters

Name	Type	In/out	Description
pCallLeg	SipCallLeg*	In	Handle of Call Leg on which request is received.
pSipMsg	SipMsg*	In	Handle of SIP Message that is received.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

3.1.2. HandleSIPResponse

MsRetcode HandleSIPResponse(SipCallLeg pCallLeg, SipMsg* pSipMsg);*

Purpose

This function is used handle SIP responses received by Media Server.

Parameters

Name	Type	In/out	Description
pCallLeg	SipCallLeg*	In	Handle of Call Leg on which request is received.
pSipMsg	SipMsg*	In	Handle of SIP Message that is received.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

4.1 Conference Processing APIs

4.1.1. HandleConferenceRequest

MsRetCode HandleConferenceRequest(SipCallLeg pCallLeg , SipMsg* pSipMsg);*

Purpose

This function handles SIP Request received on a conference URI.

Parameters

Name	Type	In/out	Description
pCallLeg	SipCallLeg*	In	Handle of Call Leg on which request is received.
pSipMsg	SipMsg*	In	Handle of SIP Message that is received.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

4.1.2. HandleConferenceResponse

MsRetCode HandleConferenceResponse (SipCallLeg pCallLeg , SipMsg* pSipMsg);*

Purpose

This function handles the conference response.

Parameters

Name	Type	In/out	Description
pCallLeg	SipCallLeg*	In	Handle of Call Leg on which request is received.
pSipMsg	SipMsg*	In	Handle of SIP Message that is received.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

4.1.3. CreateConference

MsRetCode CreateConference (string StrConfId, stConfigureConf pConfigConf);*

Purpose

This function is used to create a new conference.

Parameters

Name	Type	In/out	Description
StrConfId	string	In	Unique ID from the conference from Request URI.
pConfigConf	stConfigureConf *	In	Parameter for configuring conference.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

4.1.4. DestroyConference

MsRetCode DestroyConference (string StrConfId);

Purpose

It removes the conference.

Parameters

Name	Type	In/out	Description
StrConfId	string	In	Unique ID from the conference from Request URI.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

4.1.5. AddParticipant

MsRetCode AddParticipant (string StrConfId, SipCallLeg pCallLeg);*

Purpose

This function adds the new participant in the conference.

Parameters

Name	Type	In/out	Description
StrConfId	string	In	Unique Id for the Conference from Request URI.
pCallLeg	SipCallLeg*	In	Call Leg handle to add in a conference.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

4.1.6. RemoveParticipant

MsRetCode RemoveParticipant (string StrConfId, SipCallLeg pCallLeg);*

Purpose

This function removes participant from the conference.

Parameters

Name	Type	In/out	Description
StrConfId	string	In	Unique Id for the Conference from Request URI.
pCallLeg	SipCallLeg*	In	Call Leg handle to add in a conference.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

4.2. Non Conference Processing API

4.2.1. HandleNonConferenceRequest

MsRetCode HandleNonConferenceRequest (SipCallLeg pCallLeg, SipMsg* pSipMsg);*

Purpose

This function handles non- conference based requests.

Parameters

Name	Type	In/out	Description
pCallLeg	SipCallLeg*	In	Handle of Call Leg on which request is received.
pSipMsg	SipMsg*	In	Handle of SIP Message that is received.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

4.2.2. HandleNonConferenceResponse

MsRetCode HandleNonConferenceResponse (SipCallLeg pCallLeg, SipMsg* pSipMsg);*

Purpose

This function handles response of non – conference based requests.

Parameters

Name	Type	In/out	Description
pSipCallLeg	SipCallLeg*	In	Handle of Call Leg on which call is received.
pSipMsg	SipMsg*	In	Handle of SIP Message that is received.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

5) MS Call Leg

```
MsCallLeg
{
    CallLegState           eState;
    MsMcmI*               pMscml
    SipCallLeg*          pSipCallLeg
    CallLegType*         etype
    MsCallLegMedia*     pMedia
} MsCallleg_t;
```

Structure Explanation

Fields:

eState

This represents the current state of a call.

pMscml

Handle of MSCML object associated with this leg.

pSipCallLeg

Handle of SIP Call Leg.

eType

Type of Call Leg.

PMedia

Handle of Media Objects associated with this Leg.

a. MsCallLegMode

```
Typedef enum MsCallLegMode
{
    MsAPI-SendOnly,
    MsAPI-ReceiveOnly,
    MsAPI-SendRecvOnly,
    MsAPI-Inactive,
} MsCallLegMode_t
```

b. CallLegType

```
typedef enum CallLegType  
{  
    MsAPI-ConferenceControlLeg,  
    MsAPI-UserLeg,  
  
} ConferenceControlLeg_t
```

6) MS Call Leg Event

```
MsCallLegEvent  
{  
    MsCallLegEventType      eType;  
    MsCallLegEventData*    pdata  
  
} MsCallLegEvent_t;
```

Structure Explanation**Fields:**

eType
This represents the type of event

pdata
This represents the data corresponding to type of events

6.1 User Call Leg API

6.1.1. EventProcessor

MsRetCode EventProcessor(MsCallLegEvent pMsCIEvent);*

Purpose

Processes incoming events for Call Leg according to its state machine

Parameters

Name	Type	In/out	Description
pMsCIEvent	MsCallLegEvent*	In	Pointer to event structure

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

6.1.2. SendInfo

MsRetCode SendInfo (MsMscml pMscmlMessage);*

Purpose

Send an INFO message on this Call Leg

Parameters

Name	Type	In/out	Description
pMscmlMessage	MsMscml*	In	MSCML message to send with the message

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

7) MS MSCML Event

```
MsMscmlEvent
{
    MsMscmlEventType      eType;
    MsMscmlEventData*    pdata

} MsMscmlEventType_t;
```

Structure Explanation

Fields:

eType
This represents the type of event

pdata
This represents the data corresponding to type of events.

8) MS Call Leg Event

```
MsCallLegEvent
{
    MsCallLegEventType      eType;
    MsCallLegEventData*    pdata

} MsCallLegEvent_t;
```

Structure Explanation

Fields:

eType
This represents the type of event

pdata
This represents the data corresponding to type of events

9) MS MSCML

```

MsMscml
{
    MsMscmlStMachine      pStatMachine;
    MsMscmlMessage*      pMscmlReq,

} MsCallLegEvent_t;

```

Structure Explanation

Fields:

eStateMachine
This is handle of state machine.

pMscmlReq
Handle of MSCML Request.

a. MsMscmlEventType

typedef enum MsMscmlEventType

```

{
    MS_MSCML_DTMF_KEY,
    MS_MSCML_RECORDING_STARTED,
    MS_MSCML_RECORDING_STOPPED,
    MS_MSCML_RECORDING_ERROR,
    MS_MSCML_PLAYING_STARTED,
    MS_MSCML_PLAYING_STOPPED
    MS_MSCML_PLAYING_ERROR,
    MS_MSCML_TIMER_EXPIRED,
    MS_MSCML_REQUEST,

```

}MsMscmlEventType_t

b. MsCallLegEventType

typedef enum MsCallLegEventType

```

{
    MS_CL_SIP_REQUEST
    MS_CL_SIP_RESPONSE

```

```

MS_CL_DTMF_KEY
MS_CL_MSCML_RESPONSE

```

```

} MsCallLegEventType_t

```

9.1. MSCML Processing API

9.1.1. EventProcessor

```

MsRetCode EventProcessor(MsMscmlEvent* pMscmlEvent);

```

Purpose

Processes incoming events according to its state machine.

Parameters

Name	Type	In/out	Description
pMscmlEvent	MsMscmlEvent*	In	Handle to MSMCL event.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration

9.1.2. StartTimer

```

MsRetCode StartTimer(unsigned int iIntervalMillisec, void* pTimerData, void*
pTimerId);

```

Purpose

This function starts a timer.

Parameters

Name	Type	In/out	Description
iIntervalMillisec	unsigned int	In	Interval of timer in milliseconds
pTimerData	void*	In	Handle to data to return after timer is expired
pTimerId	void*	Out	Handle to timer identifier

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration

9.1.3. StopTimer

MsRetCode StopTimer(void pTimerId);*

Purpose

This function stops a timer.

Parameters

Name	Type	In/out	Description
pTimerId	void*	In	Handle to timer identifier

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

9.1.4. SendMSCMLResponse

MsRetCode SendMSCMLResponse (MsMscmlMessage pMscmlResponse);*

Purpose

This function sends MSCML response.

Parameters

Name	Type	In/out	Description
pMscmlResponse	MsMscmlMessage*	Out	Handle of MSMCL response.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

9.1.5. Play

MsRetCode Play(pPlayNode pPlayNode);*

Purpose

This function plays a Call Leg as specified in <play>, <playrecord> and <playcollect> tag.

Parameters

Name	Type	In/out	Description
pPlayNode	pPlayNode*	In	Pointer to play node in MSCML message.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

9.1.6. Collect

MsRetCode Collect(pCollectNode pCollectNode);*

Purpose

This function collects digits on a call leg as specified in <playcollect> tag.

Parameters

Name	Type	In/out	Description
pCollectNode	CollectNode*	In	Pointer to Collect Node in MSCML message.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Structure.

9.1.7. Record

MsRetCode Record(pRecordNode pRecordNode);*

Purpose

This function records media on a Call Leg as specified in <playrecord> tag.

Parameters

Name	Type	In/out	Description
pRecordNode	RecordNode*	In	Pointer to Record Node in MSCML message.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

9.1.8. Stop

MsRetCode Stop(pStopNode pStopNode);*

Purpose

This function stops an operation on a call leg.

Parameters

Name	Type	In/out	Description
pStopNode	StopNode*	In	Pointer to StopNode in MSCML message.

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode Enumeration.

10) MS Call Leg Media

```
MsCallLegMedia
{
    CallLegMode          eMode;
    MsSdp*                pSdp

} MsCallLegMedia_t;
```

Structure Explanation

Fields:

eMode

This represents the current mode of the call leg.

pSdp

Handle of SDP objects associated with this leg.

a. CallLegType

```
Typedef enum CallLegType
{
    CLT_User,
    CLT_Type,

} CallLegType_t
```

b. CallLegMode

```
Typedef enum CallLegMode
{
    CLM_Sendonly
    CLM_Recvonly
    CLM_Sendrecv
    CLM_Inactive
} CallLegMode_t
```

10.1 Media Processing API

10.1.1. AllocateMedia

MsRetCode AllocateMedia(MsCallLegMedia pMedia);*

Purpose

Allocate resources for a media channel.

Parameters

Name	Type	In/out	Description
pMedia	MsCallLegMedia*	Out	Handle of Media object

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode **Enumeration**.

10.1.2. ConfigureMedia

MsRetCode ConfigureMedia(SDP pSdp);*

Purpose

Configure a media channel with negotiated media parameters

Parameters

Name	Type	In/out	Description
pSdp	SDP*	In	Negotiated SDP on the media channel

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode **Enumeration**.

10.1.3.UnConfigureMedia

MsRetCode UnConfigureMedia(MsCallLegMedia pMedia);*

Purpose

Remove the configuration

Parameters

Name	Type	In/out	Description
pMedia	MsCallLegMedia*	In	Handle of Media object to operate on

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode **Enumeration**.

10.1.4. UnallocateMedia

MsRetCode UnallocateMedia(MsCallLegMedia pMedia);*

Purpose

Unallocate resources for a media channel

Parameters

Name	Type	In/out	Description
pMedia	MsCallLegMedia*	In	Handle of Media object to operate on

Return value

This function will return MsRetCode, which will contain one of the possible values as defined in MsRetCode **Enumeration**.