SLOVENSKI STANDARD
SIST ES 202 915-4-5 V1.2.1:2005

01-januar-2005

Odprti dostop do storitve (OSA) ï Vmesnik za aplikacijsko programiranje (API) ï 4. del: Krmiljenje klica ï 5. poddel: Krmiljenje konferenį nega klica SCF

Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 5: Conference Call Control SCF (Parlay 4)

Ta slovenski standard je istoveten z: ES 202 915-4-5 Version 1.2.1

ICS:

33.040.01	Telekomunikacijski sistemi na splošno	Telecommunication systems in general

SIST ES 202 915-4-5 V1.2.1:2005 **en**

ETSI ES 202 915-4-5 V1.2.1 (2003-08)

ETSI Standard

Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 5: Conference Call Control SCF (Parlay 4)

Reference
RES/SPAN-120096-4-5

Keywords
API, IDL, OSA, UML

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, send your comment to:
editor@etsi.org

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 ETSI Standard (ES) has been produced by ETSI Technical Committee Services and Protocols for Advanced Networks (SPAN).

The present document is part 4, sub-part 5 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API), as identified below. The API specification (ES 202 915) is structured in the following parts:

Part 1: "Overview";

Part 2: "Common Data Definitions";

Part 3: "Framework";

Part 4: "Call Control";

Sub-part 1: "Call Control Common Definitions";

Sub-part 2: "Generic Call Control SCF";

Sub-part 3: "Multi-Party Call Control SCF";

Sub-part 4: "Multi-Media Call Control SCF";

Sub-part 5: "Conference Call Control SCF";

Part 5: "User Interaction SCF";

Part 6: "Mobility SCF";

Part 7: "Terminal Capabilities SCF";

Part 8: "Data Session Control SCF";

Part 9: "Generic Messaging SCF";

Part 10: "Connectivity Manager SCF";

Part 11: "Account Management SCF";

Part 12: "Charging SCF";

Part 13: "Policy management SCF";

Part 14: "Presence and Availability Management SCF".

The present document has been defined jointly between ETSI, The Parlay Group (http://www.parlay.org) and the 3GPP, in co-operation with a number of JAIN™ Community (http://www.java.sun.com/products/jain) member companies.

The present document forms part of the Parlay 4.1 set of specifications.

1 Scope

The present document is part 4, sub-part 5 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

The OSA specifications define an architecture that enables application developers to make use of network functionality through an open standardised interface, i.e. the OSA APIs.

The present document specifies the Conference Call Control Service Capability Feature (SCF) aspects of the interface. All aspects of the Conference Call Control SCF are defined here, these being:

- Sequence Diagrams
- Class Diagrams
- Interface specification plus detailed method descriptions
- State Transition diagrams
- Data Definitions
- IDL Description of the interfaces
- WSDL Description of the interfaces
- Reference to the Java API description of the interfaces

The process by which this task is accomplished is through the use of object modelling techniques described by the Unified Modelling Language (UML).

2 References

The references listed in clause 2 of ES 202 915-1 contain provisions which, through reference in this text, constitute provisions of the present document.

ETSI ES 202 915-1: "Open Service Access (OSA); Application Programming Interface (API); Part 1: Overview (Parlay 4)".

ETSI ES 202 915-2: "Open Service Access (OSA); Application Programming Interface (API); Part 2: Common Data Definitions (Parlay 4)".

ETSI ES 202 915-4-1: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 1: Call Control Common Definitions (Parlay 4)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 202 915-1 apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations defined in ES 202 915-1 apply.

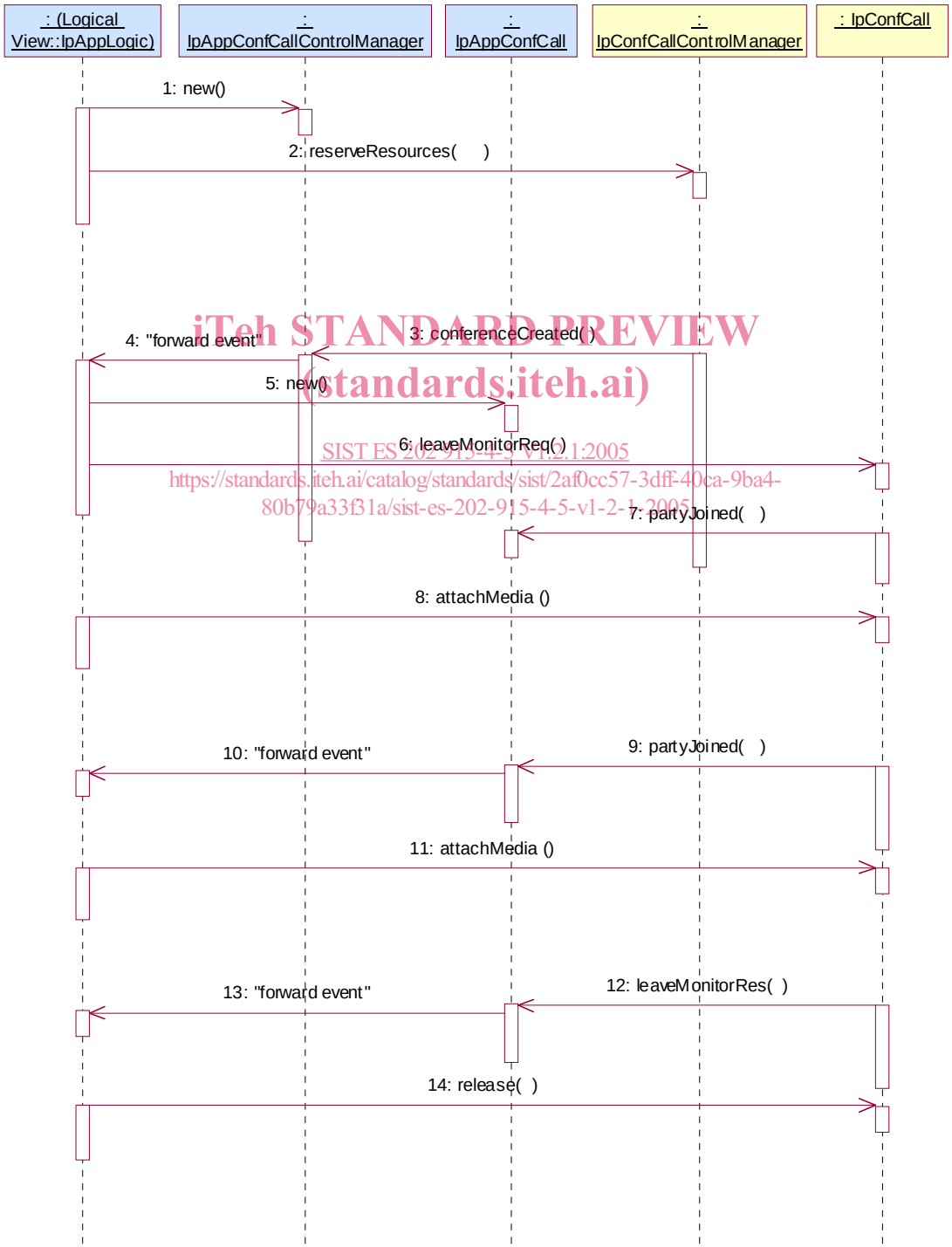
4 Conference Call Control Service Sequence Diagrams

4.1 Meet-me conference without subconferencing

This sequence illustrates a pre-arranged meet-me conference for a specified time period. During this timeslot parties can 'call in to' the meet-me conference by dialling a special number.

For each participant joining the conference, the application can decide to accept the participant in to the conference.

The application can also be notified when parties are leaving the conference.



1: The application creates a new object to receive the callbacks from the conference call control manager.

2: The application reserves resources for some time in the future.

With this same method the application registers interest in the creation of the conference (e.g. when the first party to joins the conference or at the specified start time, this is implementation dependant).

The reservation also includes the conference policy. One of the elements is whether joined parties must be explicitly attached. If so, this is treated as an implicit joinMonitorReq.

3: The conference is created.

4: The message is forwarded to the application.

5: The application creates an object to receive the call back messages from the conference call.

6: The application also requests to be notified when parties leave the conference.

7: The application is notified of the first party that joined the conference

8: When the party is allowed to join the conference, the party is added.

Alternatively, the party could have been rejected with a releaseCallLeg.

9: A new party joins the conference and the application is notified.

10: The message is forwarded to the application.

11: This party also is allowed into the conference by attaching the leg.

12: A party leaves the conference.

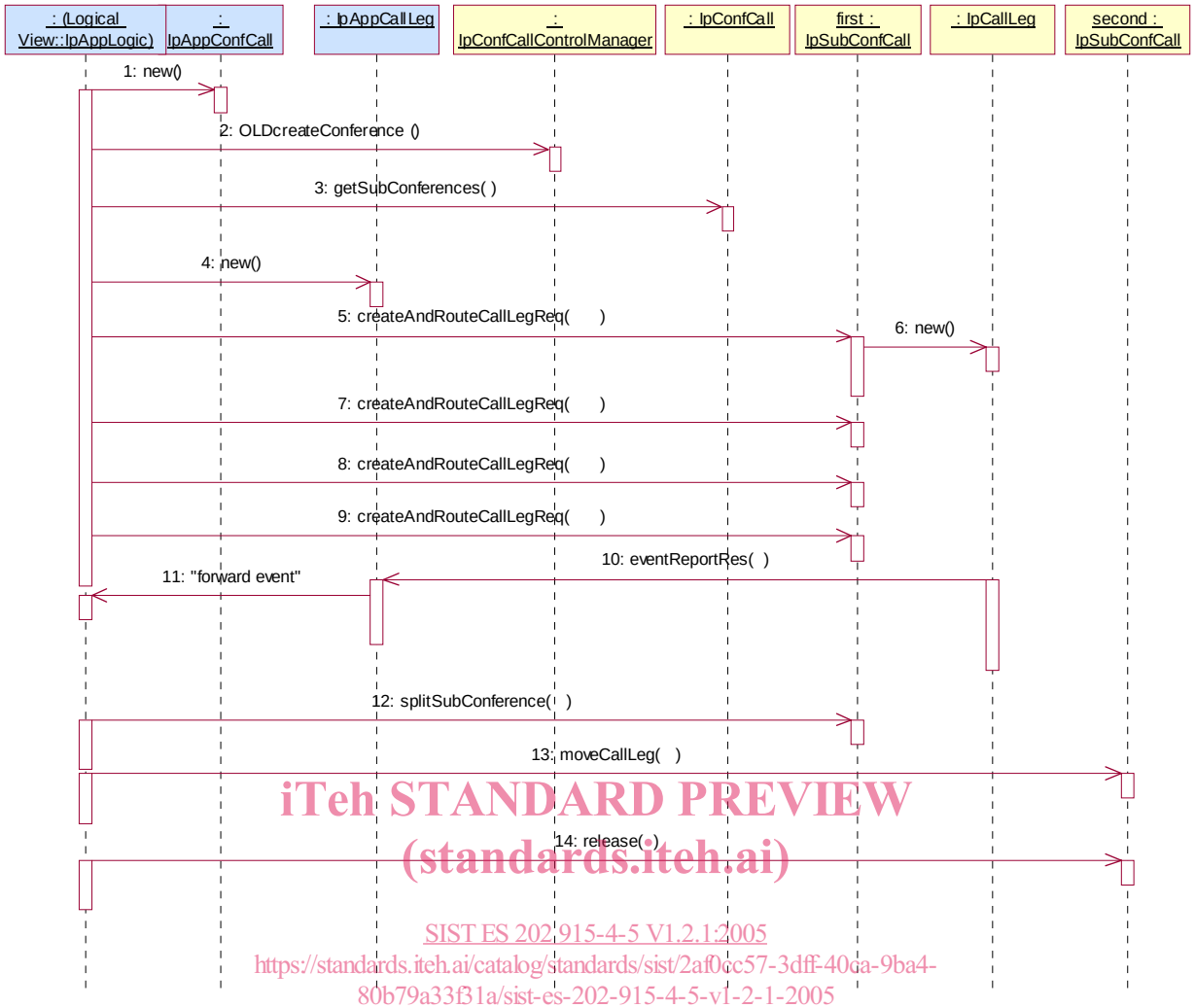
13: The message is forwarded to the application.

14: The application decides to release the entire conference.

4.2 Non-add hoc add-on with subconferencing

This sequence illustrates a prearranged add-on conference. The end user that initiates the call, communicates with the conference application via a web interface (not shown). By dragging and dropping names from the addressbook, the end-users add parties to the conference.

Also via the web-interface, the end-user can group parties in subconferences. Only parties in the same subconference can talk to each other.



1: The application creates a new interface to receive the callbacks from the conference call.

2: The application initiates the conference. There has been no prior resource reservation, so there is a chance that no resources are available when parties are added to the conference.

The conferenceCall interface object is returned.

3: Together with the conference a subconference is implicitly created.

However, the subconference is not returned as a result of the createConference, therefore the application uses this method to get the subconference.

4: The application creates a new IpAppCallLeg interface

5: The application adds the first party to the subconference. This process is repeated for all 4 parties. Note that in the following not all steps are shown.

6: The gateway creates a new IpCallLeg interface.

7: The application adds parties to the subconference.

8: The application adds parties to the subconference.

9: The application adds parties to the subconference.

10: When a party A answers the application is notified.

We assume that all parties answer. This happens in the same way as for party A and is not shown in the following.

11: The message is forwarded to the application.

12: The application decides to split the conference. Party C&D are indicated in the message.

The gateway will create a new subconference and move party C and D to the new subconference.

The configuration is A&B are in speech, C&D are in speech. There is no bearer connection between the two subconferences.

13: The application moves one of the legs from the second subconference back to the first. The configuration now is A,B&C are in speech configuration. D is alone in its own subconference.

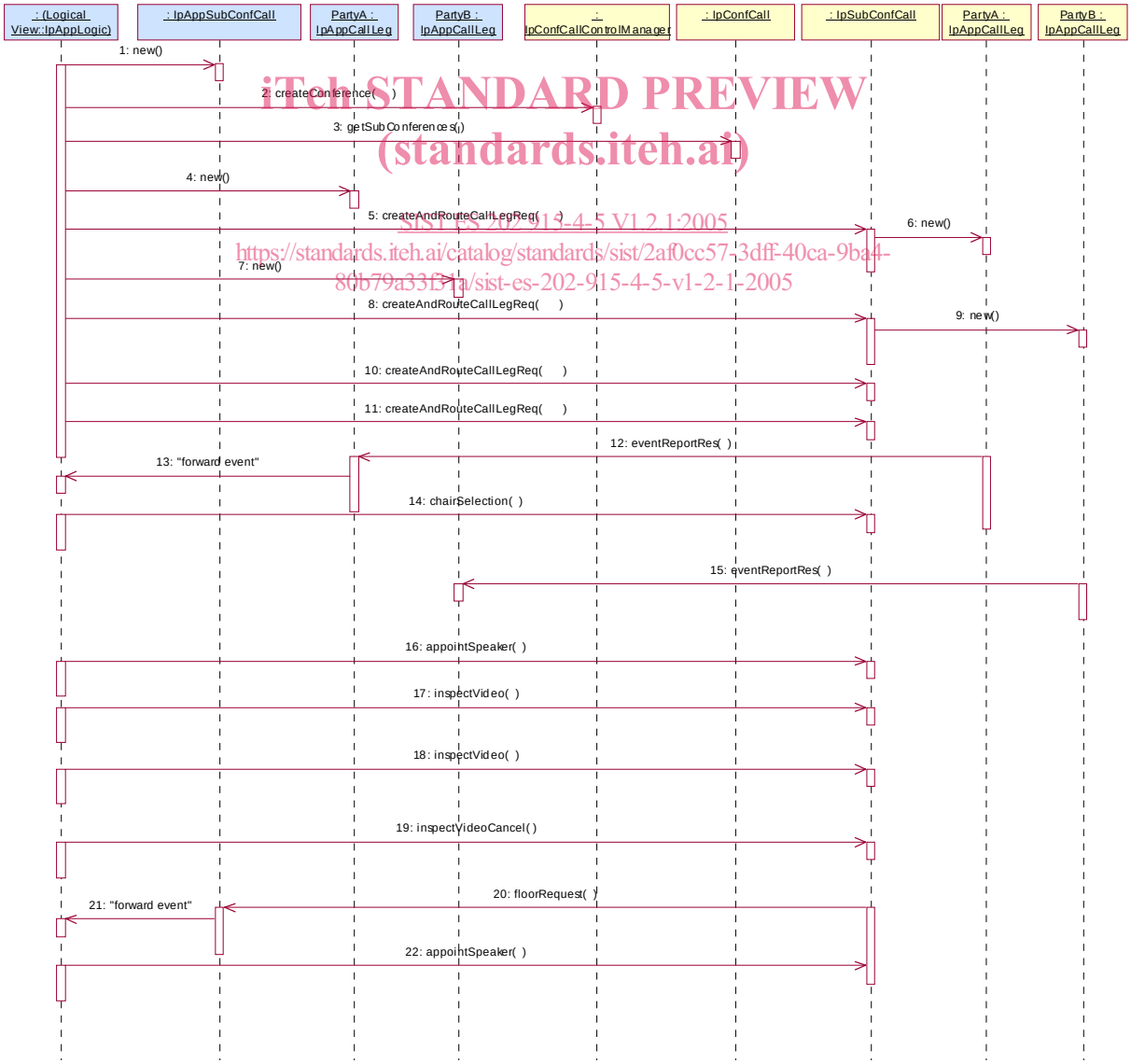
14: The second subconference is released. Since party D was in this subconference, this callleg is also released.

This leaves one subconference with A,B & C.

4.3 Non-addhoc add-on multimedia

This sequence illustrates a prearranged add-on multi-media conference. The end user that initiates the call, communicates with the conference application via a web interface (not shown). By dragging and dropping names from the addressbook, the end-users add parties to the conference.

Also via the web-interface, the end-user can do things that normally the chair would be able to do, e.g., determine who has the floor (e.g., whose video is being broadcast to the other participants) or inspect the video of participants who do not have the floor (e.g., to see how they react to the current speaker).



1: The application creates a new object for receiving callbacks from the MMSubConference.

2: When the user selects the appropriate option in the web interface, the application will create a conference without resource reservation. The policy for video is set to 'chairperson switched.

3: The application requests the subconference that was implicitly created together with the conference.

4: The application creates a new IpAppCallLeg interface.

5: The application adds the first party to the subconference. This process is repeated for all 4 parties. Note that in the following not all steps are shown.

6: The gateway creates a new IpCallLeg interface.

7: The application creates a new IpAppCallLeg interface.

8: The application adds parties to the conference and monitors on success.

9: The gateway creates a new IpCallLeg interface.

10: The application adds parties to the conference and monitors on success.

11: The application adds parties to the conference and monitors on success.

12: When a party A answers the application is notified.

We assume that all parties answer.

14: We assume that A was the initiating party.

The initiating end-user is assigned the chairpersonship.

This message is needed to synchronise the chairpersonship in the application with the MCU chairpersonship, since the chair can also use H.323 messages to control the conference.

15: When a party B answers the application is notified. We assume the other parties answer as well and this is not shown below in the sequence.

16: Chairperson (A) decides via WWW interface that party B is the speaker. This means that the video of B is broadcast to the rest.

17: The chairperson selects the video of C in order to judge their reactions on B's proposal.

18: The chairperson selects the video of D in order to judge their reactions on B's proposal.

19: The chairperson goes back to receiving the broadcasted videostream (B)

20: User C requests the floor via the H.323 signals. The application is notified of this.

21: The message is forwarded to the application logic.

22: The chairperson (via the WWW interface) grants the request by appointing C as the speaker.