TRANSITION WITHIN SIP BASED MULTI-PARTY COMMUNICATION: FROM TIGHTLY-COUPLED TO FULLY-COUPLED CONFERENCE

Wajdi Elleuch, Alain C. Houle

Université de Sherbrooke

ABSTRACT

Many models and topologies have been proposed for multi-party internet conferencing. The end-system mixing topology, where nodes are loosely coupled, is considered as one of the most popular, easy to deploy and less expensive among the other topologies. Otherwise, in some context, the centralized topology presents some drawbacks that limit conference development particularly when focus user (mixer) has to leave conference or for other cases where participant number becomes too high to be correctly mixed. On the other hand, the fully-coupled conference model spreads the mixing load between participants and enables them to leave the conference without preventing remaining participants to continue conferencing. Enabling transition to fully-coupled model should resolve tightly coupled limitations. This work explains how to enable such a transition between the two models using SIP protocol primitives. Two approaches to enable model transition are explored and their performance is evaluated.

Index Terms— multi-party conference, tightly-coupled topology, fully-coupled topology, model transition, SIP communication mobility.

1. INTRODUCTION

The Session Initiation Protocol [1], SIP, is the Internet Engineering Task Force's standard for managing multimedia sessions through signaling mechanism based on messages exchange between SIP User Agents (UA). By using SIP, UA can contact other UA to initiate, negotiate, establish and terminate communication between them. SIP provides inter alia media description, user location and registration features to support multimedia communication environment.

Since SIP protocol was considered by 3GPP as the standard for the IMS services [2], it's justified to use it as a basis for session establishment and management to provide conference services. Aside multicast conferences in which the user's media is sent to a multicast group, SIP based communications are usually established among only two participants. To enable communication for more than two-party within a SIP call, many approaches were proposed.

While the tightly model [3][4][5][6] is characterized by the simplicity of its deployment and its inherence support of two-party to multi-party transition, some limitations might restrict it use. As with all centralized approaches where service load is focused on one element called "Focus", the resulted system presents the "central point of failure" problem. For example, when Focus leaves the conference, it will not be possible for remaining participants to continue conferencing. In other scenarios, where a mobile user acts as focus using wireless access, connection bandwidth might limit the number of supported participant and QoS might then be affected. Many solutions can be proposed to reduce such problem. For example, it's possible to share conference mixing load with some other participants, or even switch from the actual focus to another one. In this work, we propose to spread the conference mixing load to all participants. To do that, full-mesh topology [7] needs to be constructed through seamless transition from tightly-coupled to fully-coupled conference model.

In the next section of this paper we give an overview about different conference models and we concentrate on the tightly and fully-coupled approaches. In section three, we define and detail the implicit and explicit approaches used to enable model transition. On the last sections, we evaluate the performance of the proposed approaches and we give conclusion to this work.

2. CONFERENCING MODELS

Many models have been proposed to enable multi-party communication for conferences. These models can be distinguished based on the topology of signaling and media relationships. The Loosely coupled model has been developed to enable large-scale multi-party communication. This model is based on the IP multicast technique where each active participant should use a multicast address that contains the addresses of participants to reach. Over and above overloading networks, this approach requires multicast-enabled routers. Therefore, deploying loosely-coupled conferences outside local networks is not conceivable and will not be considered in this work. Some other conferencing approaches use peer-to-peer signaling protocols to establish tightly coupled conferences by selecting one or more focus participants that maintain communication with other participants. Basic deployment of such an approach creates a single tree-shaped topology. That tree can be used to distribute both signaling and media (Figure 1a). This requires the tree-root (focus) to also act as

CCECE/CCGEI May 5-7 2008 Niagara Falls. Canada
978-1-4244-1643-1/08/$25.00

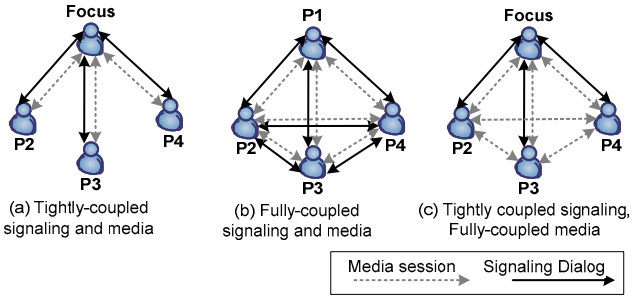


Figure 1: Conference topologies

media mixer. Managing conferences in this model will be analyzed in this section. In other models, only signaling is distributed in tightly coupled mode while media flow is fully distributed between nodes (Figure 1c). Some other conference models use the fully-coupled topology where each participant maintains both dialog signaling and media session flow with each other (Figure 1b). This fully-coupled approach enables equitable distribution of the media mixing load within an ad-hoc structure.

Since the model illustrated in Figure 1c presents the limitation of the central point of failure, the fully-coupled model of Figure 1b is most probably more reliable. In this work, we consider the fully-coupled model as a target for tightly coupled model transitions. On the remaining of this section, we focus on fully-coupled model transitions and we present related works and technical mechanisms that create and manage conferences within each model.

2.1. Tightly-coupled approach

In this approach, dialog signaling and media mixing are both carried out by one UA, the conference focus. The focus takes media from users who participate on the conference, mixes them, and sends out the appropriately mixed stream to each participant separately. Two conference models can be created depending on the resource that takes the Focus vocation. The Focus could be one of the N participants (*end-system mixing model*) or a dedicated conference server (CS) (*Server mixing model*). Creating and managing tightly coupled conferences by SIP primitives has been the topic of many previous works. Where some of them focus on the signaling aspect [4][5], others aim to treat this model from media management aspects [3]. Proposed solutions mainly concern conference creation/destroying and participant addition/removal. Different scenarios for joining conferences are detailed, as the dial-in, dial-out, scheduled and ad-hoc. Mechanisms that use INVITE [1], REFER [8] and NOTIFY [9] methods are widely deployed to enable third party to establish communication between two other users. In all scenarios, the conference needs to be uniquely identified by an URI. Focus participant can add the "isfocus" feature parameter in the contact header field so that remaining participants are aware that their communication session is a conference. Moreover, in [7], a user can add itself to the conference by using the *Join* header field when sending an INVITE to the conference URI. In such a field, the user includes information about existing dialog to join the conference. Such information might be obtained beforehand by various means like Web, E-Mail or even instant messaging.

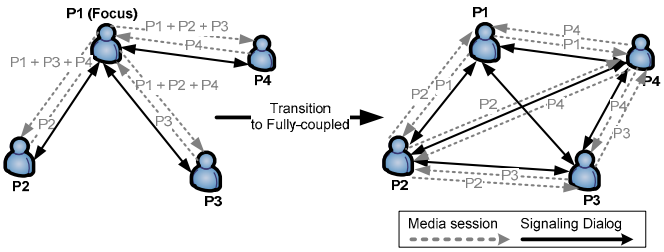
2.2. Fully-coupled approach

In this approach, every endpoint directly communicates with every other one. All the parties in the conference are "equal" and no user is topologically special or has any additional rights or abilities beyond those of the others. Any conference member can, at any time, add another user to the conference. If the new member accepts, it establishes connections to the other parties in the conference. Similarly, any member of the conference can drop out at any time, without affecting the remaining conference participants. In the fully-coupled model presented in [10], abstract protocol messages are proposed to enable conference creation and membership management. This same work shows that its abstract messages can be mapped to SIP primitives.

Implementation of the full mesh messaging protocol, as described in [10], requires introducing some new headers. Exchanged messages within the same conference need to include a *Conference-ID* header that uniquely identifies the conference. This ID should be generated by the initial conference creator, possibly using the same procedure as the one used to generate the value of the *Call-ID* field. The *Invited-By* header should be included on the INVITE message and used by the newly added participant to specify the identity of the user that invited him (the basic SIP *Contact* header can be used to identify each participant). Finally, conference members-list should be exchanged by participants and can be provided using new *Conference-Member* header field. The UPDATE [11] method can also be used to ensure conference coherence by exchanging membership information for every added/removed user.

3. FULLY-COUPLED TRANSITION

3.1. Transition approaches



Enabling transition from one conference model to another one generally require the use of mechanisms that enable creating, modifying, moving and even deleting communication dialogs between peers. Since both source and target conference topologies are based on SIP protocol, it becomes obvious to use the same protocol to realize such conference transition. Transiting from tightly-coupled to fully-coupled topology requires new dialogs establishment between all non-focus elements, as shown on Figure 2. Two approaches have been explored in this work. While the first approach (explicit approach), is based on the basic SIP REFER/NOTIFY mechanism [8][9], the second approach (implicit approach) complies with the full-mesh conference protocol detailed in [10].

3.2. Transition steps

To implement seamless transition to the full mesh model, we define, for a conference with N participants, N-1 steps. While going through the steps, focus user (P1) should proceed with one-by-one invitation to include tightly coupled users to the full mesh conference, as shown in Figure 3. After n-1 steps, P1 acts as participant on the fully-coupled conference with {P2, P3, P4…Pn} and as focus of the tightly-coupled conference with {Pn+1, Pn+2, ..PN}.

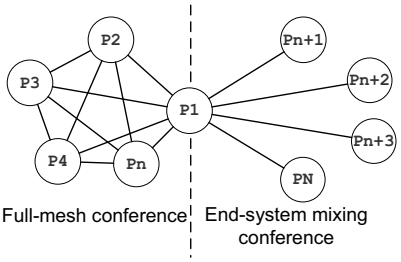


Figure 3: Conference topology after (n-1) steps

3.3. Message Protocol

To implement the required N-1 steps, each approach uses a separate messaging protocol mechanism detailed in this section. In the explicit approach, the focus element (P1) sends the SIP REFER message to the next tightly coupled participant to refer it to contact all participants added to the fully-coupled conference, as shown on the scenario on the left of the Figure 4. For a given intermediary step n, moving Pn+1 from the first group to the second group requires from the focus (P1) to send (N-2) REFER messages. For each message, P1 refers Pn+1 to contact one of the fully-coupled participants. On the other hand, for each received REFER, Pn+1 sends an INVITE to the referred participant. As soon as a positive answer is received about the invite, a NOTIFY message is sent to P1. After receiving (n-2) NOTIFY messages about all referred participants, P1 should consider that step n has been successfully accomplished and then go to the next step (n+1).

The implicit approach is quite different from the explicit approach. This approach is based on the messaging mechanism detailed in [10] and supposes that all destination nodes support extra header messages, as indicated in the protocol. To execute step n, Focus (P1) sends to the Pn+1 a JOIN abstract message that can be mapped to a SIP INVITE. In this message, P1 includes the list of all already fully-coupled participants. While receiving such message, Pn+1 will contact these participants one-by-one and by sending the CONNECT abstract message that can be mapped to the INVITE SIP message. This CONNECT

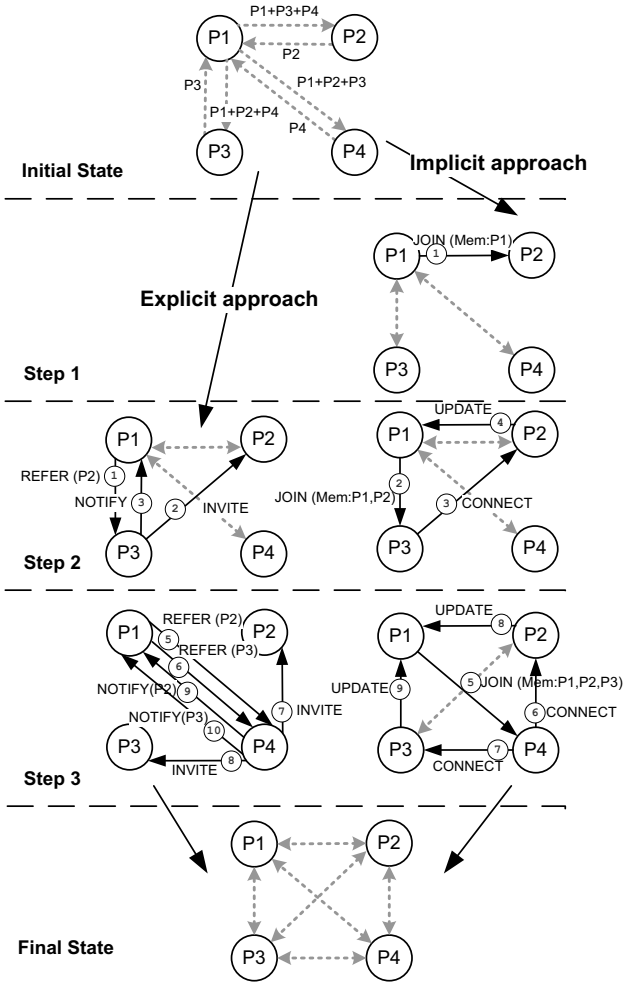


Figure 4: Full Mesh model transition steps

message includes the *Invited-by* header that specifies the address of the focus (the original inviter of Pn+1). While receiving the CONNECT message, each P user has to send an UPDATE message to notify the focus about the presence of new participant. Doing so, P1 is kept informed about the progress of dialogs establishment on the fully-coupled group and can decide whether to re-invite Pn+1 or to launch the next transition step.

4. EVALUATION OF TRANSITION MESSAGE COSTS

To evaluate transition message costs, we consider the exchanged message workload between the N participants. Since the implicit and explicit transition approaches present separate messaging mechanisms, this section includes a comparative study between them. The aim is to evaluate the message workload variation according to the initial N participant number. The general transition cost (Tcost) is expressed in (1) as the sum of all individual step costs:

$$T_{cost} = \sum_{i=1}^{N-1} (Step_{cost})i \qquad (1)$$

For the explicit approach:

$$(Step_{cost})i = (ReferT_{cost} + InviteT_{cost} + NotifyT_{cost}) * (i-1) \qquad (2)$$

For the implicit approach:

$$(Step_{cost})i = JoinT_{cost} + (ConnectT_{cost} + UpdateT_{cost}) * (i-1) \qquad (3)$$

In (2) and (3), we express step cost using transaction cost. For example, "$ReferT_{cost}$" represents the cost of the REFER transaction. Since REFER is based on the request/response mechanism, its cost is equal to 2. NOTIFY and UPDATE transactions require also 2 messages, while JOIN, CONNECT and INVITE transactions require 3 messages. In this evaluation, we suppose that all requests will have positive answers and the transaction succeeds at the first attempt.

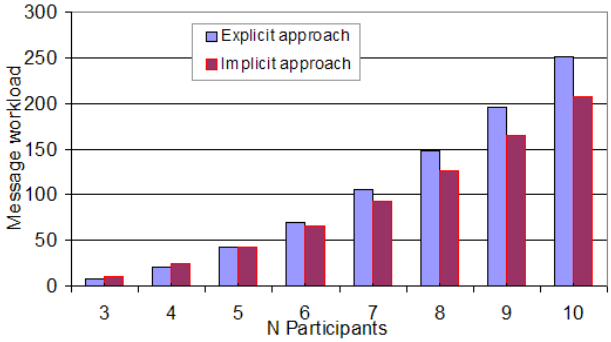


Figure 5: Message workload evaluation

From Figure 5, it can be seen that when N is equal to 5, the two approaches have an identical cost. For conference with 3 or 4 participants, explicit approach is less expensive. Otherwise, for more than 5 users, the implicit approach requires fewer messages. The cost difference between the two approaches becomes more important as number of participants increases.

5. CONCLUSION

In this work, we explored application-level solution based on SIP primitives that describe two approaches for transiting from tightly-coupled to fully-coupled conferencing model. The aim was to avoid tightly-coupled conference drawbacks. Other solutions might also be proposed to do that. As an example, spreading mixing load or even delegating this load to another available participant might be considered as serious solution paths. Theses solutions should be studied to evaluate their performance and to compare them to our solution.

6. REFERENCES

[1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Sparks, R.,Handley, M., Schooler, E., "SIP: Session Initiation Protocol", IETF RFC3261 (2002)

[2] 3GPP: TS 23.228: "IP Multimedia Subsystem (IMS) (Stage 2) -Release 5", 2002, ftp://ftp.3gpp.org/Specs/archive/23_series/23.228/

[3] Singh,K,. Nair,G., Schulzrinne,H., "Centralized Conferencing using SIP", Dept. of Computer Science/Columbia University, New York, 2001

[4] Johnston, A., Levin, O., "Session Initiation Protocol (SIP) Call Control – Conferencing for User Agents", IETF RFC 4579, 2006

[5] Rosenberg, J., "A Framework for conferencing with the Session Initiation Protocol (SIP)", IETF RFC 4353, 2006

[6] Levin, O., Even, R., "High-Level Requirements for Tightly Coupled SIP Conferencing", IETF RFC 4245, 2005

[7] Mahy, Petrie, "The Session Initiation Protocol (SIP) Join Header", RFC 3911, October 2004

[8] Sparks, R., "The Session Initiation' Protocol (SIP) Refer Method", RFC 3515, April 2003.

[9] Sparks, R., "The Session Initiation Protocol (SIP) Referred-By Mechanism", RFC 3892, September 2004.

[10] Jonathan Lennox, Henning Schulzrinne, "A Protocol for Reliable Decentralized Conferencing", International Workshop on Network and Operating System Support for Digital Audio and Video, 2003

[11] Rosenberg, J., "The Session Initiation Protocol (SIP) UPDATE Method", IETF RFC 3311, 2002