Scalable Distributed Conference Control in Heterogeneous Peer-to-Peer Scenarios with SIP*

Alexander Knauf & Thomas C. Schmidt
HAW Hamburg, Dept. Informatik
Berliner Tor 7
D–20099 Hamburg, Germany
Alexander.Knauf@haw-hamburg.de,
t.schmidt@ieee.org

Matthias Wählisch
Freie Universität Berlin,
Institut für Informatik
Takustr. 9
D–14195 Berlin, Germany
waehlisch@ieee.org

ABSTRACT

An increasing number of popular conferencing applications operate in a lightweight, infrastructure-independent ad hoc fashion and extend into the mobile realm. These P2P-type systems raise the demand for scalable, adaptive self organization of conferencing in a standard-compliant way. This paper addresses the challenge of distributed conference management with SIP and makes the following two contributions. First, we define distributed operations of a conference focus by splitting the role of identifier and locator of the conference URI. Second, we extend the SIP conference event package by states that ensure a uniformly consistent view at the conference and facilitate resource-adaptive self organization.

Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Applications—*SIP*; C.2.4 [**Distributed Systems**]: Distributed applications—*Conferencing*

General Terms

Mobile multimedia communication

Keywords

Tightly coupled SIP conferencing, distributed conference control, ID locator split, P2P systems, conference event states

1. INTRODUCTION

The design of conferencing in a fully distributed, infrastructure-independent fashion is a promising direction in both, research and practical development. Aiming at efficient and flexible communication, it faces the challenge of devising self organization for spontaneously arriving peers, and at the same time to adapt to heterogeneous capabilities at devices and network access. Demands for adapting to unbalanced communication within groups significantly grow in the presence of mobile parties.

The widely adopted standard for conference call and media management throughout the Internet is given by the Session Initiation Protocol (SIP) [1]. While SIP is inherently a peer-to-peer protocol, current multimedia conferencing solutions mostly rely on an infrastructure of central controllers. Tightly coupled conferences in SIP are defined with the help of one common, routable URI that is represented by a single focus node. In this paper we report on early work of fully distributing such a conference focus, while sustaining mutual awareness, SDP negotiations and standard compatibility. Complementary to our work on videoconferencing for mobiles [2], we introduce a solution of separating the unique conference identifier from its locators to distributed instances. Furthermore, we propose a distributed conferencing event state extension that ensures a uniformly consistent view and facilitates resource-adaptive self organization.

The remainder of this paper is organized as follows. In the next section 2 we summarize the key problems, as well as the related work. Section 3 introduces the concepts and SIP call flows for distributing a conference focus. Event state extensions for distributed conference control are presented in section 4, followed by an initial evaluation in the subsequent section 5. Finally, in section 6 we conclude the paper and give an outlook.

2. KEY PROBLEMS & RELATED WORK

A mobile peer-to-peer conferencing application faces the simultaneous challenges to assemble sufficient operational resources among peers, and to remain robust with respect to the infrastructure and its overlay peering system. It is unlikely that a single node can contribute exhaustive service as a focus and media relay throughout the conference. Instead, the role a user agent is able to attain in a distributed scenario needs to be adaptively determined according to constraints of its device and current network attachment, but also according to infrastructural hindrances as inherited from NATs and firewalls.

Traditional architectures for tightly coupled conferences rely on a single focus entity which is addressable by a globally routable conference URI [3]. This URI attains the simultaneous role of an identifier of the conference and of its locator. The major task in distributing this focus lies in split-

*This work is supported by the German Bundesministerium für Bildung und Forschung within the project *Moviecast* (http://moviecast.realmv6.org).

Mobimedia'09 September 7–9, 2009, London, UK.

Digital Object Identifier: 10.4108/ICST.MOBIMEDIA2009.7436
http://dx.doi.org/10.4108/ICST.MOBIMEDIA2009.7436

ting the functions of identifier and locator. Even though all conference members are required to logically join the same conference, they physically attach to different instances of the focus located at distinct peers. With a distributed conference control, the requirement for a consistent view of conference states arises. All focus instances need to possess an identical view about conference members and distribution states that may aid resource-adaptive self organization. Changes at one focus instance need to automatically trigger updates at all other instances.

SIP conference management standards commonly follow the centralized approach. Alternatively, a standard design for loosely coupled conferences was formulated based on multicast, which does neither foresee a mutual awareness of conference members, nor initial SDP [4] negotiations. Little work has been accomplished in the direction of tightly coupled, distributed management, as it bears an inherent complexity: Splitting a focus poses the requirement of defining a meaningful mapping from the conference URI to the group of focus instances. Cho et al. [5] have defined such a mapping in the form of a focus hierarchy. A primary focus represents the conference URI and serves as initial contact, as well as a load dispatcher. The concept of cascading conference focuses has been recently also brought into IETF [6]. However, group conferencing in these approaches remains bound to a central entity, and thus does not comply to robustness constraints of a pure peer-to-peer paradigm. Aside from conferencing, there are strong activities in the P2PSIP working group to move SIP proxy functions to a structured peer-to-peer layer [7, 8]. A distributed hash table is envisaged to aid user location and point-to-point session management. This early work has not touched the more intricate topic of group communication at the present time.

3. CONFERENCING WITH A DISTRIBUTED FOCUS

3.1 Initiation of a Conference

Ad hoc conference control is based on SIP user agents which provide an application module to create conferences on demand. A two-party dialog may change to a multi-party conversation by a third party call arriving at one of the user agents. According to RFC 4579 [9], the callee will generate a conference-specific URI and advertise it to the newly arriving caller. In this scenario, the creator of the conference URI becomes its central point of control, the focus, and maintains signaling relations to each participant.

In distributed conference control, a (primary) focus is prepared to discover other SIP user agents that support the functions for distributing the focus. The primary focus discovers a secondary focus, if it reaches its maximum number of joining participants. The secondary focus is then responsible for participants delegated from the primary focus or other secondary focuses, respectively. A user agent connected to a secondary focus remains unaware of this distributed conference control. New participants will be invited by the operating focus on behalf of the peer. The technique to perform distributed operations transparently will be explained in the next section, where we will describe a standard-compliant mechanism to discover peers.

3.2 Focus Discovery

In a peer-based conference system, it is likely to reach a threshold at a focus that indicates the maximal number of serviced participants. It can be assumed that a conference focus implementing the distributed conference control is aware of its own capacity. A basic approach to detect another user agent which may operate as secondary focus is to check on currently connected participants (see Figure 1). This is done by sending an event notification subscription [10] including the distributed-conference-info+xml event header and package. If the user agent does not support this package, it will respond with a SIP 489 Bad Event. In this case, the requesting focus proceeds with the next peer in its SIP dialog list until a peer supporting the package will be found. If no suitable participant is around, incoming calls may be declined. Otherwise, the requested user agent replies with a 200 OK and thus becomes a secondary focus for this conference. Thereafter it should subscribe to the distributed-conference event package as described in section 4. In this way, any overloaded focus is able to redirect further join requests for the conference.

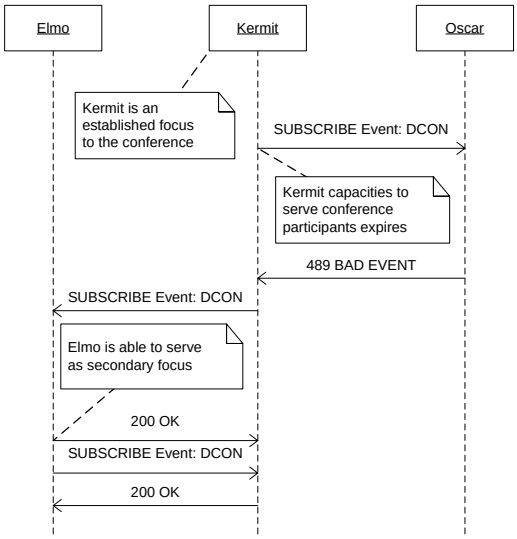

Figure 1: Discovery of a Secondary Focus

3.3 Distributing the Conference Control

We now describe the distributed operations among participants following the example displayed in Figure 2. A conference-aware participant (Oscar) would like to invite a new peer (Ernie), but the established focus (Kermit) is fully booked. Oscar's refer to Kermit is delegated to the previously discovered secondary focus Elmo. Elmo is now responsible for inviting the user agent declared in the SIP refer-to header field. If successful, he notifies the referring focus and updates all focus points which have subscribed to his distributed conference event package.

Another scenario is illustrated in Figure 3. It shows the user agent Grover, which has learned the conference URI by non-SIP means. Using this URI, Grover sends an INVITE request to an established, but fully occupied focus. This focus (Kermit) will temporally accept the request by sending a SIP 200 OK response, but delegates the call to a known secondary focus (Elmo) using a SIP REFER request to in-

Digital Object Identifier: 10.4108/ICST.MOBIMEDIA2009.7436
http://dx.doi.org/10.4108/ICST.MOBIMEDIA2009.7436

itiate a re-INVITE with the participant. Elmo accepts the request and issues an INVITE to Grover, using the record-route header mechanism explained below.

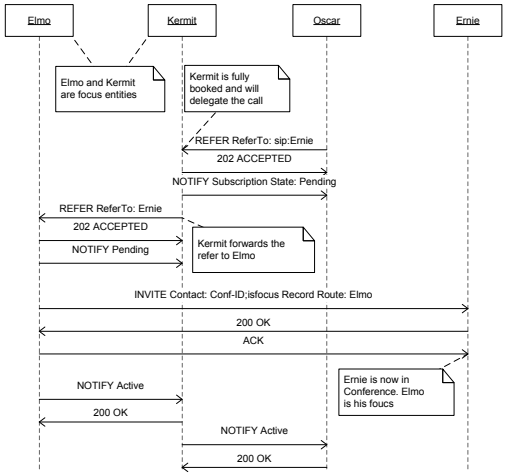

Figure 2: Delegating a call to a Secondary Focus

The decision to delegate calls to a secondary focus depends on node capacities. In contrast, accepting these calls solely relies on the secondary focuses. A metric for the initiation of distributed conferencing can be individually defined by each node and beyond the scope of this work. This scheme does not depend on individual topological structures or routing schemes. The general SIP message flow between distributed focuses and their conference participants is illustrated in Figure 4.

Backward Compatibility via Record-Route.

User agents unaware of our proposed mechanism can also participate in a conference distributed over several focuses. Deploying multiple focus entities introduces the problem of a globally routable unique URI (GRUU). A conference URI cannot be shared by multiple user agents without using an anycast address mechanism, which has to be supported by the Internet infrastructure. Therefore, every secondary focus adds a record-route header as defined in RFC 3261 [1] with the GRUU of the secondary focus.

```
INVITE ernie@Sesamestreet.com SIP/2.0
CSeq: 815 INVITE
...
Contact: <sip:conf.kermit@Sesamestreet.com>;isfocus
Record-Route: <sip:elmo@sesamstreet.com>
```

The record-route header field forces every SIP client to route further SIP requests via the secondary focus. On reception of such requests, the secondary controller will intercept the messages and process them transparently to the conference participant.

3.4 Resilience Against Focus Failure & Leave

In contrast to centralized solutions, distributed conferencing operates without an infrastructure component that guarantees a stable and always available management service. Our scenario targets at spontaneously created conferences

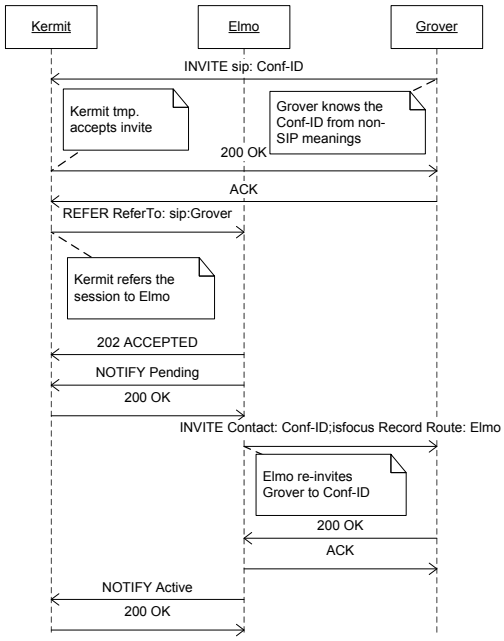

Figure 3: Joining a conference with distributed focus

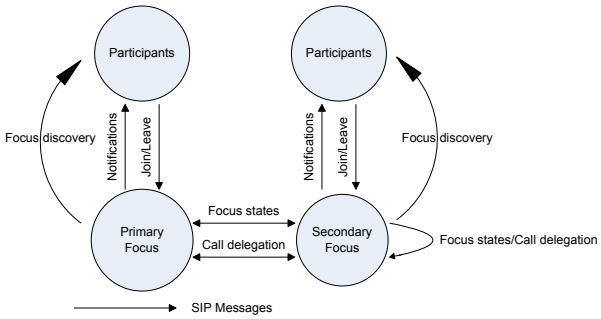

Figure 4: Message flow in a distributed conference control

between consumer computers and mobile devices like smartphones and PDAs. In such scenarios, leaves or failures of a focus device require fixing at low complexity.

At a graceful leave, the focus must delegate its established SIP dialogs to other, potentially available devices by sending SIP REFER [11] requests for each of its peers. Subsequently, the leaving focus needs to terminate its event subscriptions by refreshing the SIP SUBSCRIBE requests with expiration headers set to zero. All focus entities are now aware of the focus departure and must no longer delegate SIP calls to it.

If a focus detects the failure of another focus, it should re-INVITE all conference participants connected to the absent controller. Note that participants are known from the distributed conference event package. If needed, it may delegate calls to other controllers and arrange a load balancing. A variety of solutions for load balancing already exist, but are not discussed in this paper.

Digital Object Identifier: 10.4108/ICST.MOBIMEDIA2009.7436
http://dx.doi.org/10.4108/ICST.MOBIMEDIA2009.7436

4. THE DISTRIBUTED CONFERENCING EVENT EXTENSION PACKAGE

Tightly coupled SIP conferences commonly provide conference state information by the conference event package [12] defined within the SIP events framework. Distributing conference control not only requires one to maintain a coherent state of the distributed logic of the focus entities, but also relies on knowledge about the splitting into multiple focus entities in question, into load balancing and scalability enhancements. To provide the latter, we encode the corresponding information within a multifocus extension for the conference event package.

The aim of the extension is to inform all conference focus entities about the state and load of conference participants connected to individual controllers, the willingness of a focus to serve new participants, and to display the established SIP routes within the conference topology. These information are bound to one specific group conference identified by its unique conference URI.

Figure 5 gives an overview of the multifocus extension set encoded in XML. A focus state information document begins with the root element `<focus-states>` of focus-states-type which has to be placed as a child element `<conference-info>` from defined in [12]. It is comprised of `<conference>` and `<focus>` child elements.

Each `<focus>` element of focus-type with the cardinality `0..*` identifies exactly one conference focus. It uses the attribute key `entity` from the event package for conference state [12] to reference to the `<user>` element of user-type. The `<conf-id-holder>` of boolean type indicates if focus entity is the primary focus and is the owner of the conference URI, or denotes a secondary focus otherwise. The `<focus-capacity>` element is a container for the `<max-participant>` and `<max-focus-references>` of type int. First, this informs about the number of participants a focus is able to handle. The second element describes how many direct subscriptions of remote focuses are allowed to this distributed conference controller. Thereby every focus is able to publish his maximal capacity to the conference and avoid requests that would cause overload.

The `<participants>` element is the list of SIP user agents which are connected to this focus. It uses the attribute key `entity` of [12] to refer directly to the `<user>` element. The `<next-hops>` element is a container for the `<ref-to-focus>` element of type anyURI which identifies a single focus entity. This information makes SIP routes transparent that have been established, and may be used to initiate new direct overlay links in case the hop distance to some focus extends „too far“. It uses also the attribute key `entity` to refer to the `<user>` element.

Using this extended event package, each focus will learn about all other controllers along with its load capacities and the direct subscription routes within the conference. This information can be used to delegate calls to other secondary focus by making a lookup in the conference state xml. A method `get delegate` may retrieves the SIP URI of the lowest loaded secondary focus. Like in [12], the multifocus extension set is defined for permitting partial notification using the `state` attribute in the root element `<focus-states>`.

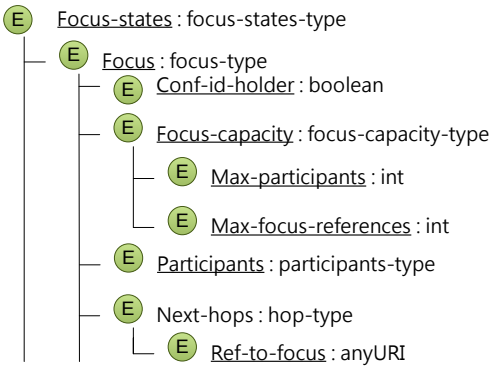

Figure 5: Distributed Conference Event Package Extension

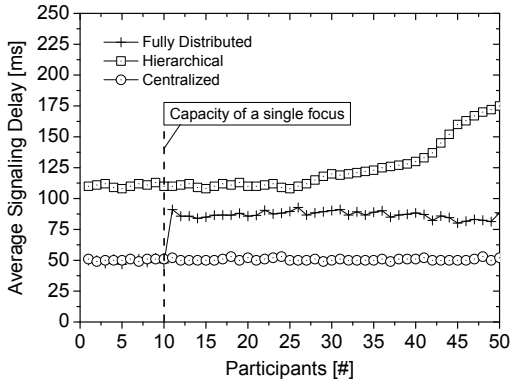

Figure 6: Performance of Signaling Delays in Distributed Conference Invite

5. EVALUATIONS

The scheme was implemented using the NIST Jain SIP stack [13], and first evaluations were performed in an emulation mode. Figure 6 displays the measurements of signaling delays in comparison with a centralized approach and the hierarchical scheme of [5]. The capacity of a single focus was fixed to 10 conference members, and delay values were obtained from 30 independent runs.

For small conferences, where all parties can be served from a single focus, our results agree with delays of the centralized approach. The redistribution of the focus attachment in our scheme causes one additional REFER message and thus almost doubles the signaling times. Apart from this delay enhancement, the distributed conferencing admits constant delays, in contrast to the hierarchical scheme. The latter experiences increasing delays of approximately linear scale with growing conference size. Further evaluations, in particular the gains from adaptive media streaming and error resilience will be analyzed in forthcoming work.

6. CONCLUSIONS & OUTLOOK

Enhancing scalability and error resilience of SIP conferencing is a major objective. In this paper, P2P approaches to distributing conference control have been identified to address the problem in a transparent, standard compliant fashion. We presented a corresponding SIP protocol scheme that allows for spontaneous and infrastructure-independent

Digital Object Identifier: 10.4108/ICST.MOBIMEDIA2009.7436
http://dx.doi.org/10.4108/ICST.MOBIMEDIA2009.7436

conferencing of a large number of heterogeneous participants, and at the same time are confined to constant signaling overhead.

In future work, we will elaborate this early development, extend our analysis and optimizations with the aim of including this scalable conference management scheme into our mobile videoconferencing solution [2].

7. REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," IETF, RFC 3261, Jun. 2002.

[2] H. L. Cycon, T. C. Schmidt, G. Hege, M. Wählisch, D. Marpe, and M. Palkow, "Peer-to-Peer Videoconferencing with H.264 Software Codec for Mobiles," in *WoWMoM08 – Workshop on Mobile Video Delivery (MoViD)*, R. Jain and M. Kumar, Eds., Piscataway, NJ, USA: IEEE Press, June 2008, pp. 1–6.

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

[4] J. Rosenberg and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)," IETF, RFC 3264, Jun. 2002.

[5] Y.-H. Cho, M.-S. Jeong, J.-W. Nah, W.-H. Lee, and J.-T. Park, "Policy-Based Distributed Management Architecture for Large-Scale Enterprise Conferencing Service Using SIP," *Selected Areas in Communications, IEEE Journal on*, vol. 23, no. 10, pp. 1934–1949, Oct. 2005.

[6] S. Romano, A. Amirante, T. Castaldi, L. Miniero, and A. Buono, "Requirements for Distributed Conferencing," IETF, ID – work in pr. 04, Dec. 2008.

[7] D. Bryan, P. Matthews, E. Shim, D. Willis, and S. Dawkins, "Concepts and Terminology for Peer to Peer SIP," IETF, ID – work in progress 02, Jul. 2008.

[8] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, "REsource LOcation And Discovery (RELOAD)," IETF, Internet Draft – work in progress 00, Jul. 2008.

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

[10] A. B. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification," RFC 3265, Jun. 2002.

[11] R. Sparks, "The Session Initiation Protocol (SIP) Refer Method," IETF, RFC 3515, Apr. 2003.

[12] J. Rosenberg, H. Schulzrinne, and O. Levin, "A Session Initiation Protocol (SIP) Event Package for Conference State," IETF, RFC 4575, Aug. 2006.

[13] "The NIST JAIN-SIP homepage," http://jain-sip.dev.java.net/, 2009.

Digital Object Identifier: 10.4108/ICST.MOBIMEDIA2009.7436
http://dx.doi.org/10.4108/ICST.MOBIMEDIA2009.7436