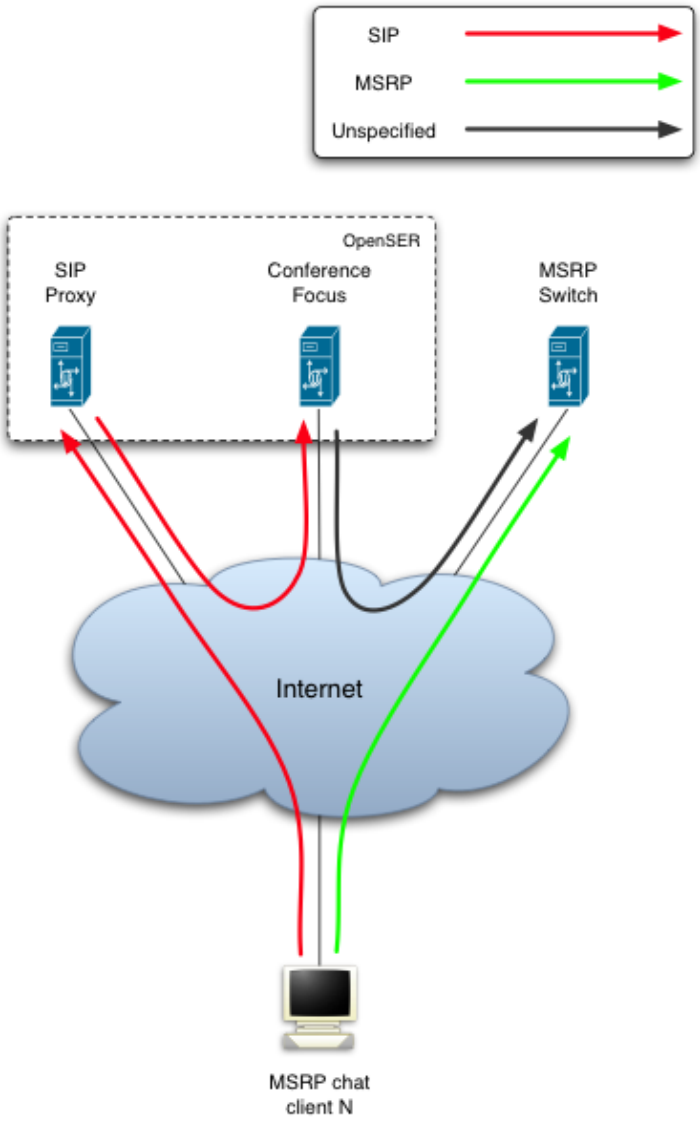
# MSRP conference design

> This is the original version of the design that has been replaced with the use of SIP SIMPLE client. The current design and the implemented software based on it, is available at http://chatserver.ag-projects.com

## Components

The image below identifies the different separate components involved in a SIP/MSRP chat session. Note that only one chat client is shown, while in practice an arbitrary number of clients will participate in the chat session.



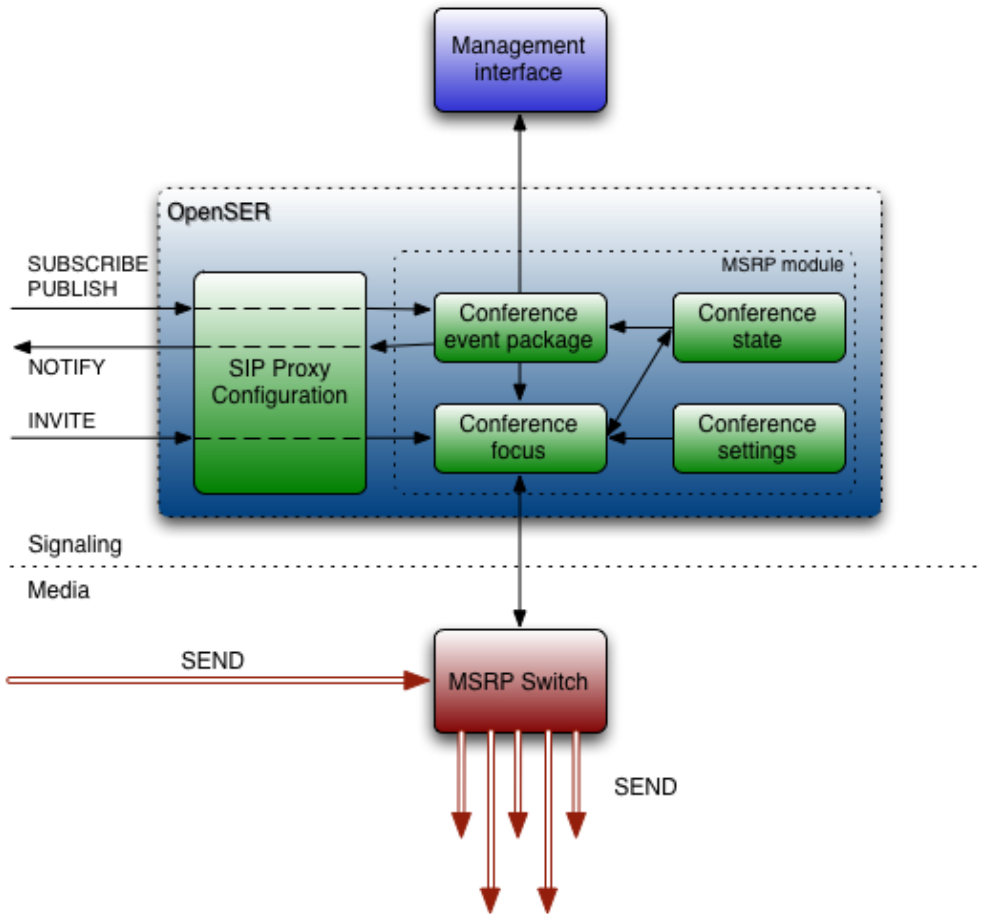

The clients communicate session state through the SIP proxy with the "conference focus", an entity which is defined within the scope of SIP conferencing ( RFC 4353). As can be seen in the diagram, the conference focus will be integrated within OpenSER.

The clients sends and receives MSRP messages to and from the MSRP switch, which effectively mixes all MSRP traffic of the clients within the chat session.

The conference focus will need to communicate with the MSRP switch about various aspects of the chat session. This communication will be done through an unspecified protocol, the primitives of which will be identified later in this document through message flow examples.

# Architecture

The conference focus will be implemented as an OpenSER extension module. This or a secondary module will also implement a subset of the "conference event package", as defined in RFC 4575, supporting such things as chat participant identification.



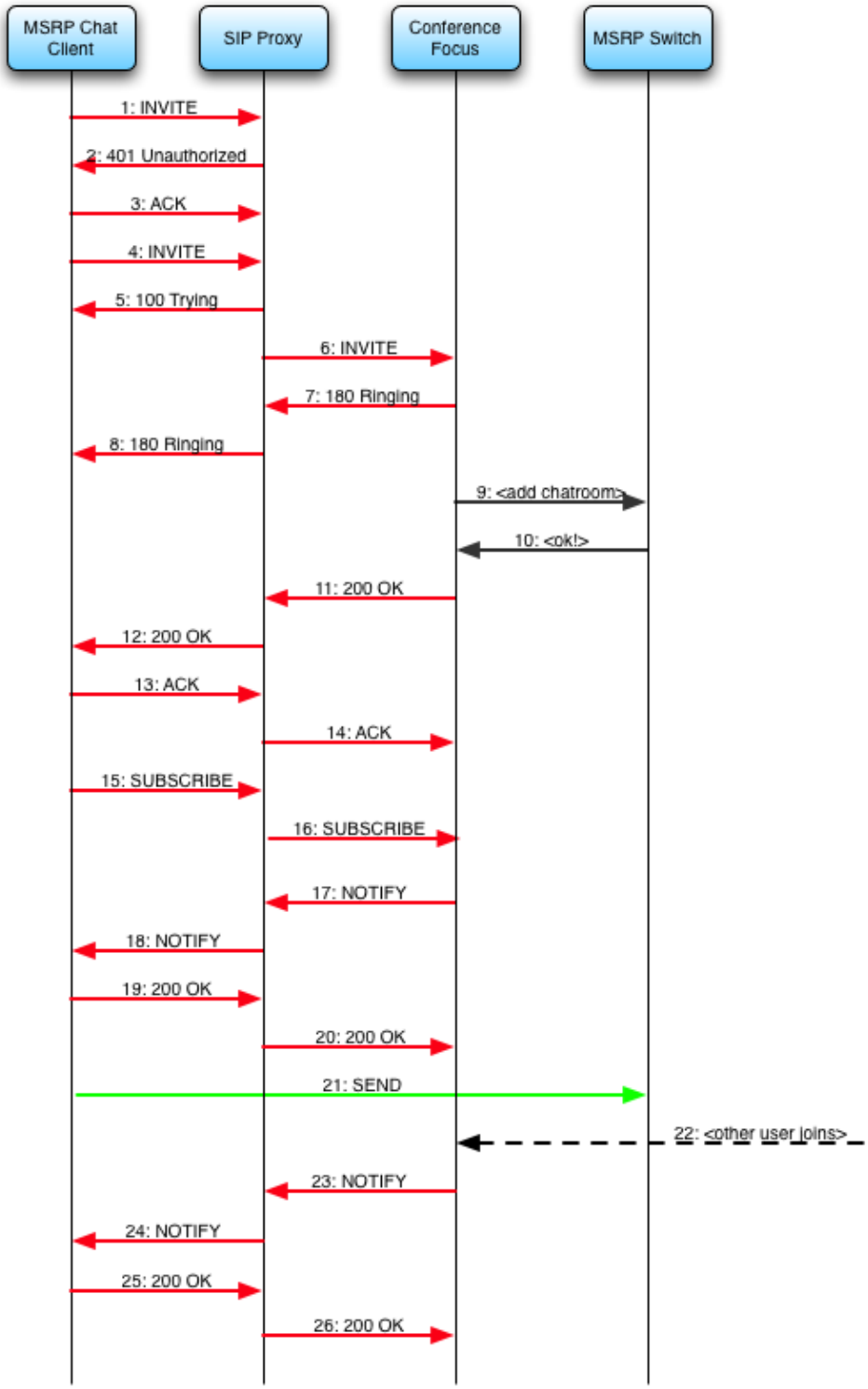

The conference focus functionality functions as a SIP endpoint within the SIP proxy, communicating with the clients through the SIP INVITE, REFER and BYE methods, while the "conference event package" module receives SUBSCRIBE/PUBLISH methods from the server and the clients and sends out NOTIFY.

Also pictured is the communication with the MSRP switch and a management interface, both of them through some unspecified protocol.

# Message flows

This section contains four examples of message flows between the entities identified within the previous sections.

## Client joins a chat




A client sends an INVITE to the conference focus of a particular MSRP chat room. This could be a room that exists already, or one that it requests be created. The SIP proxy identifies the target SIP URI as belonging to a MSRP chat conference, possibly through some preconfigured hostname such a chatroomX@conference.example.com, and redirects the INVITE to the conference focus module. This module first checks local configuration and state to see if this SIP URI belongs to a chat conference that already exists, either a preconfigured room or an ad-hoc created conference. If this is not the case it also

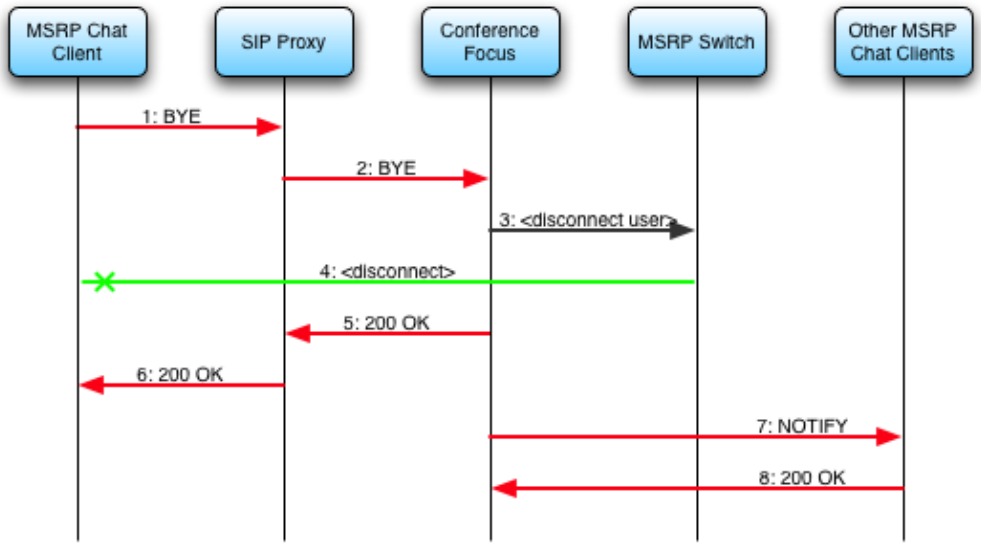
checks local policy to see if the client is allowed to create a chat conference in an ad-hoc manner. If all is well it will pass the request to the MSRP switch, which will send some reply as to wether this client can join the chat conference. If this is the case, as it is in this diagram, the conference focus sends the client a "200 OK" message, including in the SDP the media information of the MSRP switch, and the normal SIP INVITE negotiation continues.

Once the client has completed the INVITE it will SUBSCRIBE to the conference state event for this particular conference SIP URI. The received state in the NOTIFY will include at least information on which other clients are currently in the chat conference. The client is then ready to send and receive MSRP traffic to and from the MSRP switch.

The diagram also shows that if another client joins (or leaves) the chat conference, this will be indicated to each client that subscribed to the conference event package through a NOTIFY. This notify will only contain a delta of that which was contained in the initial NOTIFY to the client.

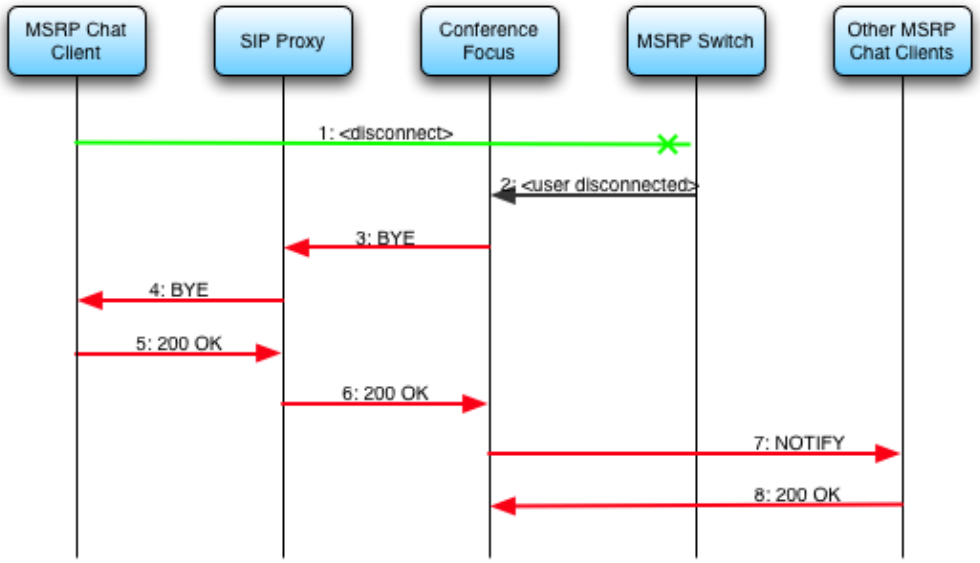
Optionally, the client may indicate parties to be invited to the conference by attaching a resource-list document to the INVITE. On receive of such body, the server will trigger REFER to the participants indicated in the resource-list attachement.

## Client leaves through BYE



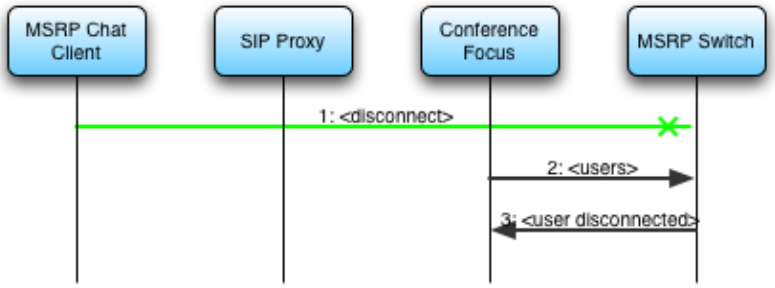

A client can leave the chat session by sending a BYE to the conference focus. This will then signal the MSRP switch to disconnect the MSRP TCP connection to this client. It will also inform those clients that subscribed to the conference state that this user is no longer participating in the chat session.

## Client disconnects from MSRP switch




If the client for some reasons does not send a BYE, but simply disconnects its TCP connection from the MSRP switch, or if the TCP connection to this client is timed out, the MSRP switch needs to indicate this to the conference focus, which in turn will inform the subscribed clients that this user has left the chat conference. For sake of completeness it can also send a BYE to the client that disconnected.

## OpenSER restart




In order not to destroy active conferences when OpenSER is restarted, it can save its conference state to a persistent database. When OpenSER comes back online it uses this state to query the MSRP switch for any changes in the conference while it was down. As the only change that can occur at the MSRP switch is the disconnection of one or more clients, the OpenSER module will present a list of clients known to be in a particular confernce to the MSRP switch. The switch will then inform OpenSER of any clients that are no longer in the chat conference and the OpenSER module can update its internal state to match the current situation.