## Multiparty WebRTC Communication

The goal of this document is to provide specifications for enabling WebRTC group communication in reTHINK (aka WebRTC Multiparty). Furthermore, an ongoing discussions about these specs can be found here [WebRTC Multiparty #106](https://github.com/reTHINK-project/dev-runtime-core/issues/106#issuecomment-245019063).

We can distinguish two topologies :

### Multiparty Topologies

In full mesh WebRTC conference, every peer establishes a connection with the rest of the participant peers. Thus, n\*(n-1) number of connections are needed, where n is the number of participant peers. For examples, a full mesh topology of 4 users has 12 connections.

This topology performs well for a small number for participants. Easy to deploy without extra resources(no server is needed).

However, it is inefficient for scalable multiparty systems. As the number of participants increases, bandwidth and CPU processing consumption becomes more excessive on peer side, which is not scalable, especially for mobile Peers.



Full Mesh Topology for group communication

Alternatively, in a star topology, a relay server in the middle will be in charge of establishing peer connections and distributing streams among peers. As a result, each peer establishes only one peer connection to the media server independently of the increasing number of peers. Which is very scalable approach. In such way, all the burden and processing is left to the middle server. However, an extra latency will be observed due to the presence of the intermediary server.

This star topology, describes H2H WebRTC group communication between reTHINK users. Therefore, peer conference hyperties running on runtime browsers will exchange signaling descriptions between each other and the media server through reTHINK edge server(Runtime Node).Particularly, reTHINK Runtime Node [dev-runtime-nodejs](https://github.com/reTHINK-project/dev-runtime-nodejs) is a justified choice for exchanging WebRTC signaling. In addition, Runtime Node is a fully conform with reTHINK specs in term of reliability and security.

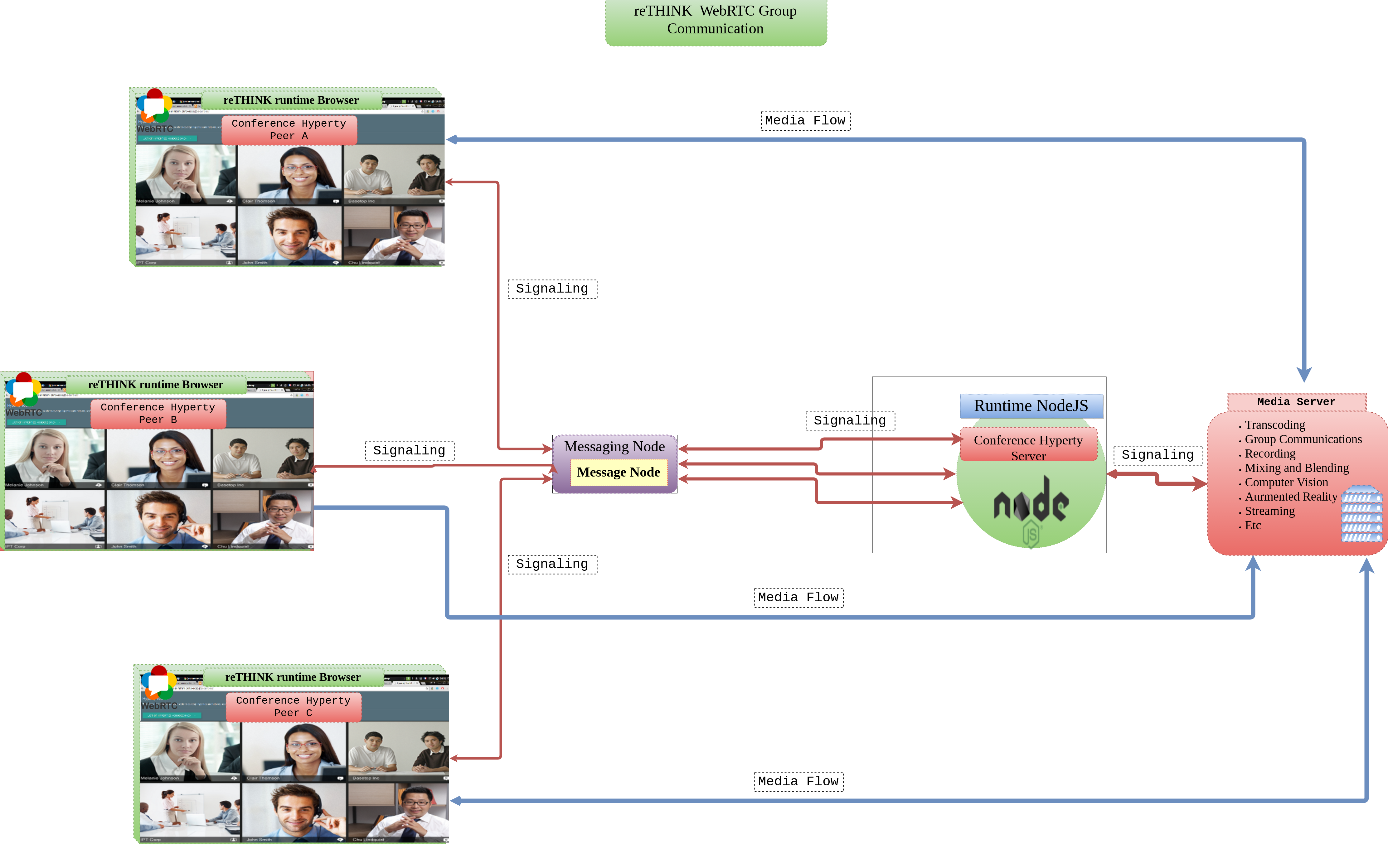


Star Topology for group communication

### Solution for Star Topologies

According to the reasons presented above, star topology seems a good candidate for scalable rethink WebRTC group communication.

Three main components are required in this topology : Peer conference Hyperty running on Runtime browser, Conference Hyperty running on Runtime Node, and the Media Server. While, the messaging node is just relay point in reTHINK framework. Thus, it's transparent for the different communication messages.



reTHINK Group cummunication overall architecture

The call flows between the components of this topology are provided [here](https://github.com/reTHINK-project/specs/blob/master/dynamic-view/group-communication/readme.md)