文章编号：2095-2163(2019)06-0132-06 中图分类号：TP399 文献标志码：A

基于 WebRTC 的视频会议系统开发

石 芮，成洪豪，孙立民
（烟台大学 计算机与控制工程学院，山东 烟台 264005）

摘 要：现行视频会议系统在实现视频会议功能时，大都需要特定的客户端软件支持，造成不同客户端之间无法进行音视频通信。针对该问题，本文设计并实现了一套基于 WebRTC（Web Real-Time Communication，WebRTC）的多人视频会议系统。该系统采用基于集中混合型视频模型实现，该模型对终端用户减少了带宽要求，并且，各终端音视频流在混流前可以进行静音压缩，增强系统灵活性。结合依托于 Kurento 的 WebRTC 作为中间流媒体服务器的支持，实现媒体路由/调度的组通信，用于多人视频通信技术。通过将系统投放到实际单位使用检测表明，该系统在多人会议方面系统运行稳定，视频画面清晰。

关键词：WebRTC；即时通信；多人视频会议；Kurento；会议模型

Development of video conference system based on WebRTC

SHI Rui，CHENG Honghao，SUN Limin
（School of Computer and Control Engineering，Yantai University，Yantai Shangdong 264005，China）

【Abstract】 When the current video conferencing system implements the video conferencing function，most of the specific client software support is required，resulting in inability to perform audio and video communication between different clients. Aiming at this problem，this paper designs and implements a set of multi-person video conference system based on WebRTC（Web Real-Time Communication，WebRTC）. The system is implemented based on a centralized hybrid video model，which reduces bandwidth requirements for end users，and each terminal audio and video stream can be muted and compressed before mixing，which enhances the system flexibility. In combination with kurento′s WebRTC as an intermediate streaming server，this technology supports media routing/scheduling group communication for multi-person video communication technology. The test of putting the system in to actural units shows that the system runs stably in the multi-person conference and the video picture is clear.

【Key words】 webRTC；instant messaging；multi-person video conferencing；kurento；conference model

0 引 言

现行用于视频会议系统的软件，大多是安装在设备上的基于 C/S 构架的独立应用程序[1]。传统视频会议功能类的软件都需要有特定的客户端软件支持，如微信、QQ、Skype。这就会造成不同客户端之间无法进行音视频通信，无疑加大了硬件端的负载。复杂的视频编解码问题、通信协议、回声消除、去噪等复杂性很高的专业性问题，是传统视频通信类软件所面临的难题[2]。而自 Google 推出 Web 实时通信技术（Web Real-Time Communication，WebRTC）以来，这种能够跨平台通信的开源技术，能够直接为 Web 浏览器提供支持语音和视频会议及数据共享的能力，无需下载专用应用软件或插件。现阶段各大浏览器都实现了对 WebRTC 的支持，Google Chrome、Mozilla Firefox、Microsoft Edge、Apple Safari 等等，不同浏览器视频通信可以相互兼容，视频会议向着跨浏览器视频通信的方向发展，使得视频会议系统更加方便、更加易于获得，以满足用户需求。凭借以上技术优势，WebRTC 即将成为新一代实时音视频通信的技术标准[3]。

本文介绍了 WebRTC 的总体架构与关键技术，阐明了 Web 应用中信令层与媒体层逻辑关系，并完善 WebRTC 应用中媒体层面实现的不足。采用 kurento 新型媒体服务器将 WebRTC 与现有的服务器模型相结合并实现其在实际系统中的应用。开发基于 WebRTC 的视频会议系统，对研究和开发基于 IP 网络的跨浏览器视频通信具有重要的价值和意义。

1 关键技术介绍

1.1 WebRTC

作为网络视频的一项新技术，WebRTC 不是服务，也不是应用程序，是一个支持网页浏览器进行实时语

作者简介：石 芮（1993-），女，硕士研究生，主要研究方向：企业信息化；成洪豪（1994-），男，硕士研究生，主要研究方向：企业信息化；孙立民（1960-），男，博士，教授，主要研究方向：企业信息化。
通讯作者：孙立民 Email：sclmsun@126.com

收稿日期：2019-10-10

音对话或视频对话的 API，WebRTC 的总体结构分为三大部分：开发者基于实际开发多媒体应用的 Webapp 层、封装有通信协议及媒体处理引擎模块的 Web API 层、由浏览器商自主实现的输入输出层[4]。

在 WebApp 层，开发者根据自己的意愿选择开发实时音视频通信的多媒体应用，开发使用的多媒体应用大多基于集成了 Web API 的浏览器。

Web API 层集成了实时音视频通信所需要的通信协议与媒体处理引擎。通信协议由承载协议和信令协议两部分组成：承载协议使用 Websocket 全双工通信协议，一次握手建立连接后便始终保持连接，在建立的全双工连接上传输数据；信令协议使用 SDP 协议与 JSEP 协议对音视频通话进行会话控制与媒体协商。SDP 消息中包含了媒体协商必需的相关参数，JSEP（Java Session Establishment Protocol）协议对音视频通话开启会话控制。

输入输出层实现音视频捕获与媒体信息流的读取，此部分由浏览器厂商根据自己的需求进行自定义。

1.2　kurento

Kurento 是一个 WebRTC 媒体服务器和一组客户端 API[5]。其完善了 webRTC 媒体处理引擎，提出一个多媒体框架，简化 Web 和智能手机平台的高级视频应用程序的开发[6]。

提供多种方法改进升级原有 WebRTC 多媒体应用程序：在原有的媒体处理技术基础上，音频编码器中使用 Opus 编码器代替传统的 iSAC 与 iLBC 编码器，其可以覆盖人类听觉系统整个范围，采样支持从 8～(4 kHz 带宽)48 kHz(20 kHz 全频)的采样率，支持固定比特率和 6～510 kbps 可变比特率，帧大小从 2.5～60 毫秒；视频编码器中使用 H.264 视频编解码器，视频传输时被组织成网络抽象层单元（“NAL 单元”），使用面向字节流格式或面向分组格式传输；视频编解码延时小于 200 ms，满足实时视频通信的低延迟特性要求。采用 MKV 格式（Matroska 容器格式）用于录制。在传输方面，kurento 可以管理标准 RTP/RTCP 流，实现网络的传输与流控等功能，SRTP/SRTCP 流为传输的数据提供加密、消息认证、完整性保证和重放保护。此外使用 Gstreamer 支持任何编解码器之间的自动媒体转码。

2　视频会议系统分析

2.1　视频会议系统的需求分析

为了实现在尽可能减少带宽使用率的前提下，对一场群组视频会议的管理，系统遵循着完备性、正确性和逻辑性的原则，并根据视频会议的具体功能和业务流程分析核心业务需求。

表 1　群组视频会议的需求分析

Tab. 1　Demand analysis of group video conferences

功能名称	功能需求
会议发起者	会议发起者需要创建相应功能的会议并对其标识；应对将要加入本场会议人员赋予相应功能角色，对其发起视频邀请；对会议过程中申请加入的请求给与处理；对会议中视频窗口进行选择能够让其发言；对与会人员列表有增、删、查、改；对本场会议有开始、结束等权限。会议发起者只能唯一。
会议参与者	获取相应会议角色权限，接受会议发起者的邀请；可以通过搜索会议名称再次请求入会；接受会议发起人传来的视频流；进入会议为静音状态，在被选中后，作为整场视频的主窗口发言。会议参与者多个。
会议视频窗口	能够接受传过来视频流，显示实时画面；使用音频选择算法控制本窗口音频输出。
视频会议列表	在开启会议以后，在发起页面的视频会议列表中添加本次会议。后续加入本次会议的人员都要从会议列表中选择将要加入的会议名称请求加入。功能包括会议的制定、审批、查询、删除、查看等。

2.2　视频会议系统业务流程

根据上述具体功能分析，得出群组视频会议业务流程如图 1 所示。可以通过接受 kurento 媒体服务器发起的视频呼叫进入规定的会议；也可以通过访问 kurento 媒体服务器的 URL 进入当前会议。

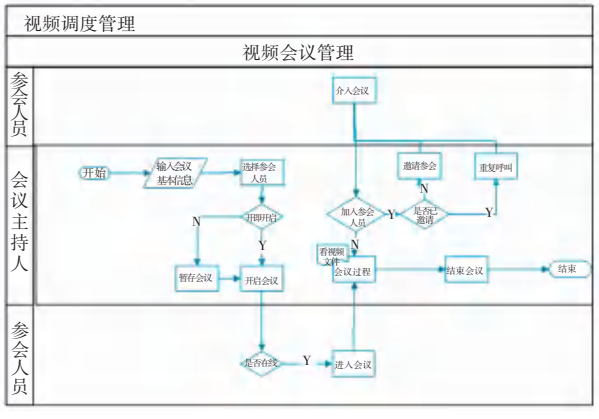

图 1　视频会议系统业务流程图

Fig. 1　Business Flow Chart of Video Conference System

3　视频会议系统模型实现

3.1　视频会议系统模型设计

视频会议模型根据信令服务器与媒体服务器对

节点的控制关系可以概括为两大类:紧耦合模式与松耦合模式[7]。紧耦合是指由一个中心节点实现信令集中控制的会议;松耦合是指无需中央 SIP 信令的控制,终端直接进行交互的会议。其中紧耦合会议又可分为端系统混合模式、集中混合模式和信令集中、媒体流分布模式。

本文中视频会议模型选择使用紧耦合会议模式下的集中混合型视频会议模型[7]。此模型中终端各成员间的通信通过一个核心的混合器来实现,每个终端成员均需与混合器建立媒体和信令的连接,每个终端只收到一个混合流,对终端用户减少了带宽要求,用户可以自由选择编码格式;音视频在混流前可以进行静音压缩,增强系统灵活性。与本文提出的服务器模式相一致,本系统中的核心混合器使用信令服务器与 kurento 服务器:信令服务器建立与各终端联系并负责对所有成员进行呼叫控制,kurento 服务器进行媒体流的混合分发。系统模型如图 2 所示。

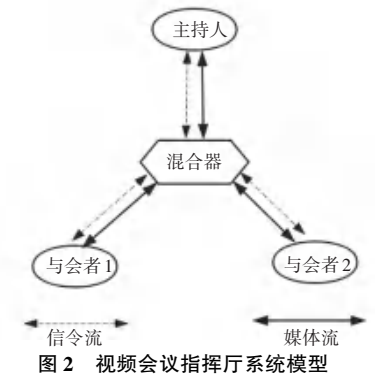

图 2 视频会议指挥厅系统模型

Fig. 2 Video conference command hall system model

模型中双向虚线代表各节点与混合器中信令服务器交互产生的信令流,双向实线代表各节点与混合器中 kurento 媒体服务器交互产生的媒体流。

3.2 模型流量控制算法设计

3.2.1 系统模型角色定义

一场会议当中的角色分为主持者(Initiator)和与会者(Participant)。每个角色在模型中都被定义为一个节点,视作模型中一个视频会议对等体。每个会议节点都能进行节点初始化、媒体流发送与媒体流接收。除此之外,主持者节点还需要发起整场会议(包括设置发起会议的名称与选择与会人员等)和负责选择切换当前窗口。

定义 一场会议为集合 $C=\{H,R_i,T,S,N_j\}$,其中 H 集合包括参与整场会议的全部节点;$R_i(i=1,2,\cdots,n)$ 集合表示为参加本次会议的与会节点,下标 i 表示为与会节点个数;T 表示为本次会议的主持者节点;S 表示会议状态(1 表示正在进行,0 表示会议未开始);$N_j(j=R_nUT)$ 集合表示当前窗口是否被选中,N_T 表示主窗口被选中,N_i^R 表示与会节点中第 i 个节点被选中。

3.2.2 算法流程

会议系统整体流程可以抽象为以下两种算法表示:使用算法 1 初始化会议中所有节点,执行会议选择算法从初始化后的节点中选择主窗口节点。

算法 1 初始化算法

Procedure:Init

Input:t 为会议主持者节点,R_k 为与会者节点集合,本次会议集合为 C

Output:

(1)BEGIN

(2)令与会者节点集合 R_k 为空

(3) *Start_listen*(t)// 初始化

(4) R_n = *req_sel_par*(t,R_k)// 主持者选择与会人员

(5)END

其中,*Start_listen* 是主持者节点初始化整场会议,包括设置会议名称、发起会议规模等;*req_sel_par* 表示与会人员通过 t 接受来自视频会议呼叫信息,t 选择与会人员集合。

算法 2 会议系统选择算法

Algorithm:select

Input:$E_k(k=1,2,\cdots,n)$ 请求与会人员列表,T 主持者节点,N_j 当前窗口状态,S 会议状态

Output:当前输出音频节点

(1) if (S == 1){// 当会议正在进行时

(2) for C_k from E_1 to E_n DO//遍历所有请求与会人员列表

(3) $R=R_nU\{E_k\}$// 将通过请求的节点加入到与会节点集合中

(4) $N_j=R_nUT$ // 窗口集合由主持者节点窗口和与会者节点窗口组成

(5) if (N_T == 1){ // 如果主持者节点窗口被选中

(6) *req_tran_media*(T) // 执行 *req_tran_media* 算法,将主窗口节点作为参数

(7) }else{

(8) if (N_i^R == 1) {//如果与会者节点窗口被选中

(9) *req_tran_media*(R_i)// 执行 *req_tran_media* 算法,将与会者节点作为参数

(10)　　}

(11)　　}

(12)　}else{

(13) *Init*()// 如果会议将要发起，执行初始化算法

(14)　　}

算法中 req_tran_media 表示当选中当前窗口时，当前窗口作为主窗口媒体流的传输，将媒体流信息分别发送到各个节点；没被选中的窗口处于静音状态，只能接受主窗口命令。

3.3　系统架构设计

根据上述关键功能的需求分析，围绕核心业务设计系统架构，如图 3 所示。整个系统架构以视频会议为驱动，采用软件架构中 MVC 多层架构设计思想搭建而成。

图 3 中信息访问层即表现层，为用户提供多种接入渠道访问本系统，保持与控制层对应关系；服务应用层实现控制层与业务层功能，使用 WebRTC 协议提供搜索引擎服务、工作流服务支持，解释用户的请求并将其映射成可执行的操作，根据具体的需求来进行业务逻辑处理；应用支持平台实现对数据访问层与数据储存层支持，kurento 服务器包含了系统中视频有关流的所有核心数据，jeecg 框架提供权限控制服务支持，用来对数据存储层的数据进行直接增、删、改、查等操作。在系统的数据传输层搭建信令服务器，使用 UDP 提供逻辑连接的建立、传输层寻址、数据传输、传输连接释放等。将通信双方需要交换的相关参数封装在 SDP 中传递给通信双方。

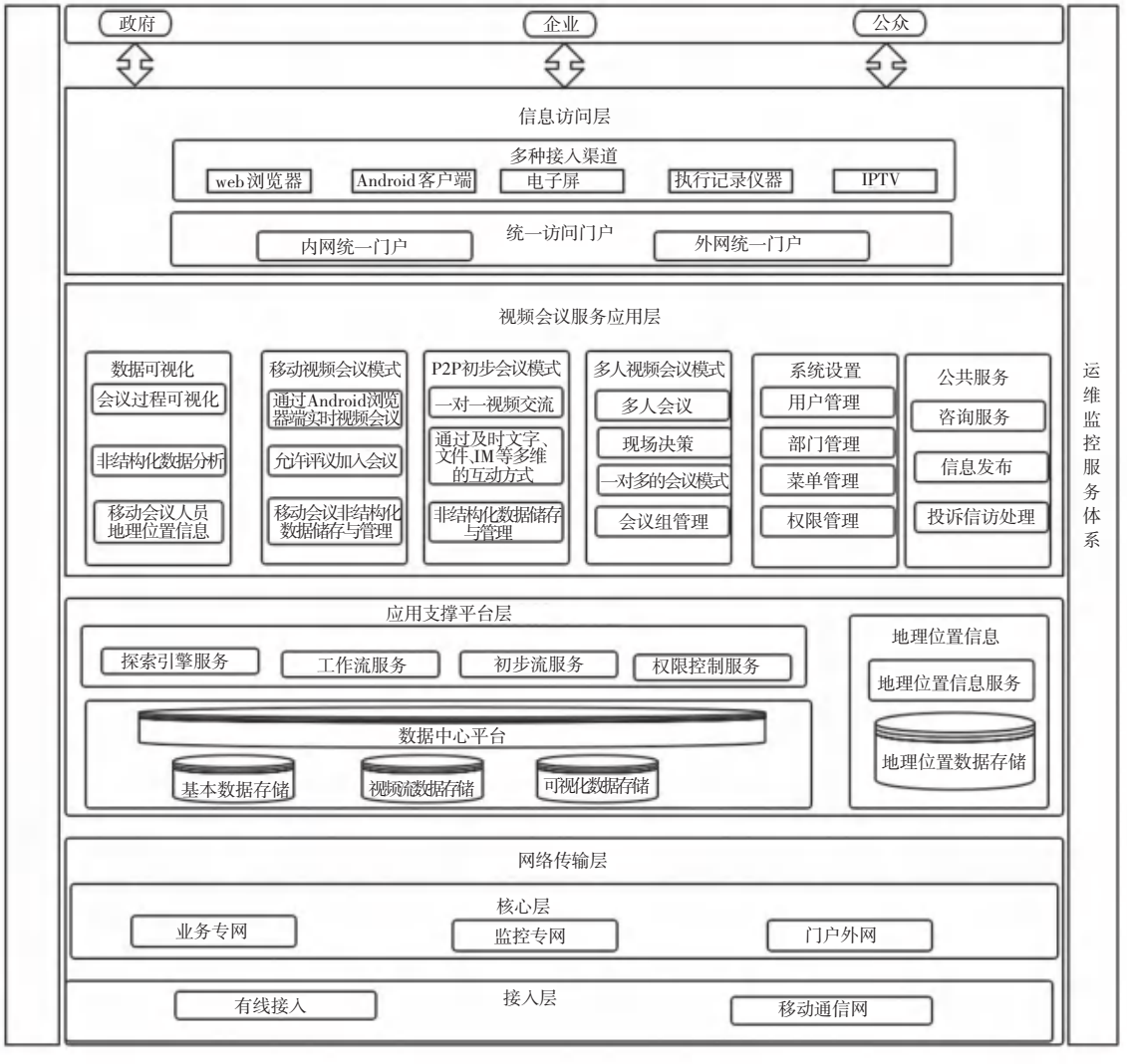

图 3　视频会议系统结构图

Fig. 3　Structure of video conference system

3.4 系统模型服务器端实现

模型中的核心混合器端由系统中信令服务器与 kurento 媒体服务器实现。每个终端均需与系统服务器端建立媒体和信令的连接。通过信令服务器与所有通信端点建立连接,负责获取初始信息并建立媒体流传输通道;kurento 媒体服务器对各个端点发送来的媒体流混合处理,将其发送到各个端点。每个终端仅会收到一个混合的流,减少了计算复杂性;通过使用 kurento 媒体服务器中的 GStream 多媒体库,对来自各个终端的编码需求处理,终端可以自由选择编码格式;并对除主持人外的与会者的音频流使用端点静音算法压缩处理,减少了带宽的使用率,系统灵活性增强,并使会议可以接受不同网络带宽性能的多样终端端点参与。系统服务器端通信架构如图 4 所示。

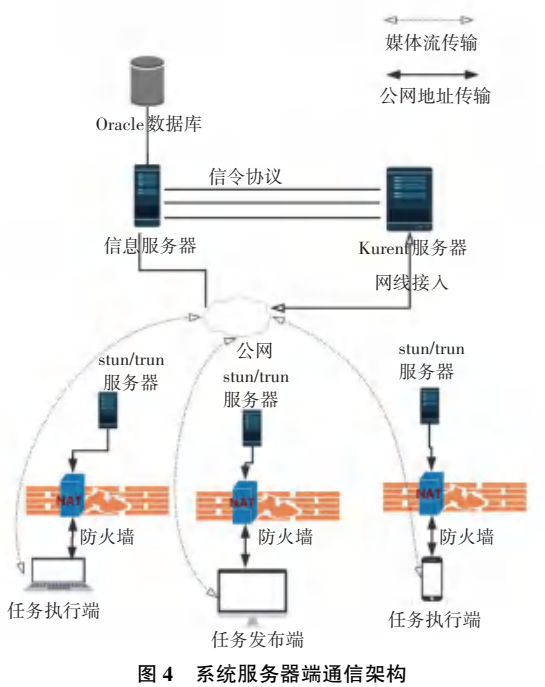

图 4 系统服务器端通信架构

Fig. 4 System server-side communication architecture

3.4.1 信令服务器搭建

信令服务器主要用于会话控制与媒体协商[8],具体的信令实现过程是基于 WebRTC 提供的弱信令 API - JSEP (JavaScript Session Establishment Protocol,JavaScript 会话建立协议)来实现。媒体协商在接到目标用户所在浏览器或终端传来的 SDP 协议格式消息以后开始,调用 JSEP 信令 API 的 createOffer ( )/createAnswer ( ) 接口,生成 SDP (Session Description Protocol),为通信双方交换的会话描述内容提供会话描述格式,保存通信双方需要交换的相关参数,SDP 消息中包含了媒体协商必需的相关参数;由浏览器调用 WebRTC 内置 API 实例化 RTCPeerConnection 对象获得自我会话描述并使用内置函数 localDescription 将其保存,通过信令通道发送到另一客户端,由此完成交换会话请求和应答消息,从而完成通信双方会话的创建。

媒体协商成功建立链接后进行媒体流传输[9],通过信令消息对音视频通话开启会话控制(会话开始、结束、信息修改等)。会话控制也调用 JSEP (Java Session Establishment Protocol)协议,其没有定义具体的实现协议,由开发者在开发时自行选择即可。此系统中使用 SIP 协议实现。

3.4.2 Kurento 服务器搭建流程

视频会议模型中核心服务器使用 kurento 媒体服务器进行处理媒体流。通过 Websocket 全双工通信建立 kurento 客户端与 kurento media serve 公开的 API 之间的连接,使用 kurento 提供的 kurento 协议来对客户端与 kurento media serve 之间的通信提供不同类型的请求/响应消息,以完成 kurento 客户端对 kurento media serve 执行的操作。

(1)首先建立客户端与 kurento media serve 之间的 websocket 链接。使用 kurento 协议中提供的 ping 方法,作为客户端发送方发送的链接方法,发送给接收方。

(2)创建用于传输的 kurento 媒体对象。使用 kurento 协议中 create 消息创建,在 create 消息中的 type 参数中指定创建媒体对象类型,如媒体管道和媒体元素等;接收方接受到发送方的 create 消息后,返回 sessionID 参数,用于创建下一步的创建请求。

(3)创建对象完成,对创建完成的媒体对象执行相应的操作。kurento 协议中 invoke 消息是用于定义要执行的操作,参数 operation 用于定义将要执行操作的名称。例如 connect 进行连接操作等。

(4)与会者需要订阅主持者发布的视频流。执行 kurento 协议中 subscribe 消息 ,参数 id 定义为主持者 id,type 参数定义订阅视频流类型,主持人端收到订阅消息后,触发 onEvent 事件,从服务器端请求视频流数据。使用 unsubscribe 消息取消对主持者端的订阅。

(5)会议结束后释放不使用的对象。使用 release 消息用于释放不需要的对象及其资源。

4 结果展示

4.1 系统开发环境搭建

PC 端使用 jeecg 面向学习型的开源 Java EE 开

发框架，在 spring-boot 核心框架基础上搭建的一个 Java 基础开发平台，使用 Vue 作为编写展示层框架实现浏览器端 HTML/js 页面，使用 MyBatis 技术实现数据访问层、控制层及业务逻辑层使用 spring-boot 框架操作，ApacheShiro 为权限授权层，Ehcahe 对常用数据进行缓存，thymeleaf 做为模板引擎，数据存储层使用 Oracle 大型数据库等 MVC 多层架构的设计模式。

4.2　视频会议系统应用场景

本模型的应用实例是税务总局对下属十六个县市区税务局召开视频直播会议[10]，要求各下属税务局逐个进行工作情况汇报。

在如图 5 所示的界面中，由税务总局主持人（会议发起者）选择需要邀请的用户列表权限，后台对数据库中所选中的邀请用户进行角色权限搜索遍历，用户角色权限符合，对其发起视频会议呼叫。

图 5　视频会议指挥系统邀请界面

Fig. 5　Video conference command system invitation interface

用户接受呼叫邀请进入视频会议界面，待所有用户都加入到会议中后，主会议页面使用响应式栅格化方式，自动分配每个用户窗口和屏幕大小，（各分局页面为主持人端与本客户端）如图 6 所示。

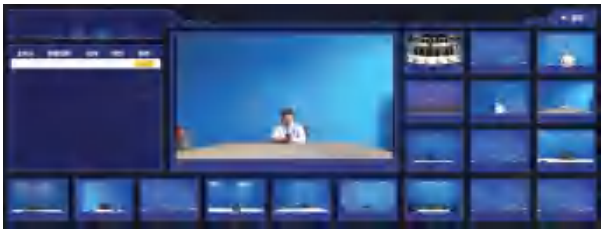

图 6　视频会议指挥系统会议界面

Fig. 6　Video conference command system conference interface

窗口则需要执行静音算法，即不对当前进行的汇报产生干扰，也减少对网络带宽的使用。经过实际测试表明，本会议模型可建立基于 P2P 的 16 路对等视频会议。在会议时长持续 24 小时中，会议过程视频流畅，无卡顿现象。

5　结束语

WebRTC 是基于浏览器的实时通信，是指运行在浏览器上的 Web 应用通过调用浏览器提供的 API，实现浏览器之间实时通信连接的建立和音视频等数据的传输[11]。本文基于这种新型的直播技术提出一种视频会议系统模型，改变传统的视频会议全连接模式，由主持人端发起会议，接受其余全部与会人员视频流信息；与会人员只需接受主持人端视频，减少服务器端负荷；对除发言窗口外其余窗口使用静音算法来减少带宽使用。根据视频会议模型设计的视频会议系统，并应用到税务部门实际工作中，进行反复检测用户加入数量对视频会议带宽影响。下一步工作中，对现有的静音算法优化，降低多用户状态下切换目标时产生的延迟。

参考文献

[1] 李兴盛. 基于 WebRTC 的实时通信系统的研究与实现[D].

[2] TIBERKAK A, LEMLOUMA T, BELKHIR A, et al. A Novel Approach for Generic Home Emergency Management and Remote Monitoring[J]. Software Practice and Experience, 2017, 48(1).

[3] 李宇轩. 基于 WebRTC 的即时通信视频系统的设计与实现[D]. 2016.

[4] 梁艳. 基于 HTML5 的 WebRTC 技术浅析[J]. 信息通信技术, 2014(2):52-56.

[5] 张向辉，黄佳庆，吴康恒，等. 基于 WebRTC 的实时视音频通信研究综述[J]. 计算机科学，2015，42(2).

[6] FERNANDEZ L L, DIAZ M P, MEJIAS R B, et al. Kurento: a media server technology for convergent WWW/mobile real-time multimedia communications supporting WebRTC[C]// World of Wireless, Mobile & Multimedia Networks. IEEE, 2013.

[7] 程志君，崔学敏. SIP 视频会议系统模型研究分析[J]. 昆明冶金高等专科学校学报，2008(3):42-46.

[8] 孙建伟，陈立，王卫. 基于 SIP 协议的 WebRTC 信令研究与应用[J]. 计算机系统应用，2018(9):273-277.

[9] 彭永超，费璟昊. 基于 Webrtc 和 PWA 的视频互动直播系统[J]. 电脑编程技巧与维护，2017(2):74-76.

[10] 林鸿，王松，杨鑫，等. 基于 WebRTC 技术的应用及平台技术开发与设计[J]. 电信科学，2013，29(9)：20-25.

[11] GONCA B, ARDA K R, MURAT T. Motion-Based Rate Adaptation in WebRTC Videoconferencing using Scalable Video Coding[J]. IEEE Transactions on Multimedia, 2018:1-1.