Publishing channel bandwidth testing

Since WCS build 5.2.1972 and SFU SDK build 2.0.273 it is possible to test publishing channel bandwidth from client to server. The test uses WebRTC data channels without a stream publishing.

WebRTC connection to a server is established when entering an SFU room

```
await state.room.join(state.pc, null, {});
```

Then `BitrateTest` object becomes available

```
const bitrateTest = state.room.getBitrateTest();
```

The `BitrateTest.test()` method launches the channel bandwidth test. The method accepts a maximum test duration in milliseconds, and returns a Promise which is resolved after the test ending. The test sends a data packets via WebRTC data channel, receives a feedbacks from the server about data received successfully, and computes the data sending bitrate value. A function called on the Promise resolution receives a latest measured bitrate value in kbps

```
bitrateTest.test($("#" +
bitrateTestState.durationId()).val()).then((bitrateKbps) => {
    stateSelector.text("Test is finished, last measured bitrate: " +
bitrateKbps + " kbps");
    ...
});
```

The `BitrateTest` object allows to listen for `onStatusUpdate` event, this event handler function receives a current measured bitrate value in kbps. This allows to display a progress indicator on a test page

```
bitrateTest.setListener({
    onStatusUpdate(bitrateKbps) {
        stateSelector.text("Current bitrate: " + bitrateKbps + " kbps");
    }
});
```

The full SFU Bitrate Test example function code to launch a channel bandwidth test

code

```
const startBitrateTest = async function (state) {
    if (state.room) {
```

```
        await state.room.join(state.pc, null, {});
        const stateSelector = $("#" + state.currentStateId());
        stateSelector.attr("style", "display:inline-block;margin-left:
10px");
        try {
            const bitrateTest = state.room.getBitrateTest();
            state.setBitrateController(bitrateTest);
            bitrateTest.setListener({
                onStatusUpdate(bitrateKbps) {
                    stateSelector.text("Current bitrate: " + bitrateKbps + "
kbps");
                }
            });
            bitrateTest.test($("#" +
bitrateTestState.durationId()).val()).then((bitrateKbps) => {
                stateSelector.text("Test is finished, last measured bitrate:
" + bitrateKbps + " kbps");
                state.setBitrateController(null);
                onStopClick(state);
            });
        } catch (e) {
            if (e.type === constants.SFU_ROOM_EVENT.OPERATION_FAILED) {
                onOperationFailed(state, e);
            } else {
                console.error("Failed to start bitrate test: " + e);
                setStatus(state.errInfoId(), e.name, "red");
                onStopClick(state);
            }
        }
    }
}
```