`replaceTrack`

To fail, or to renegotiate,

Jan-Ivar Bruaroey - mozilla

What we agree on. Instead of:

```
// A world without replaceTrack: Sender and track inextricably linked.

function replacer(pc, sender, track) {
  pc.removeTrack(sender);                 // beware the oddly named method
  var newSender = pc.addTrack(track); // new replacement sender
  return new Promise(resolve => pc.onnegotiationneeded = resolve)
    .then(() => renegotiate())            // full renegotiation
    .then(() => newSender);
}

replacer(myPeerConnection, oldSender, newTrack)
  .then(sender => log("Success!"), e => log("Replace failed: " + e));
```

What we agree on. We will have:

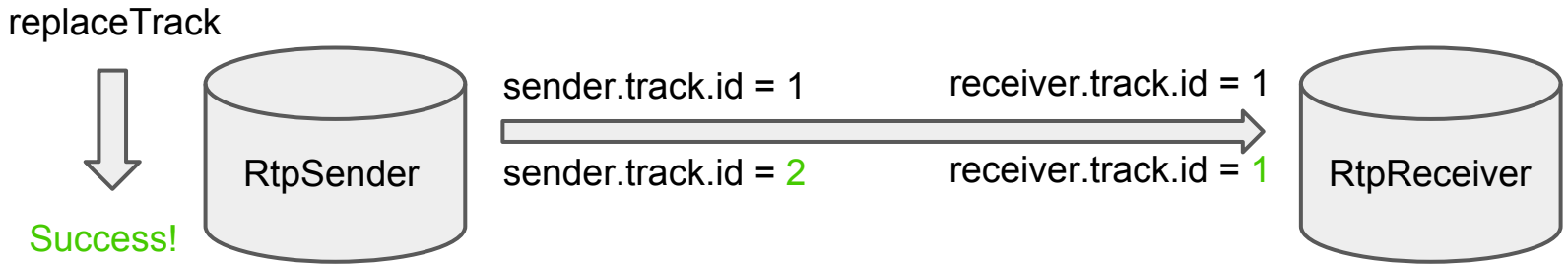
```
// replaceTrack: sender and track are separable.

var replacer = (sender, track) => sender.replaceTrack(track);
```

```
replacer(oldSender, newTrack)          // oldSender intact
  .then(() => log("Success!"), e => log("Replace failed: " + e));
```

What we agree on. This means:

```
sender.replaceTrack(newTrack).then(() => log("Success!"), e => log(e));
```

No renegotiation needed

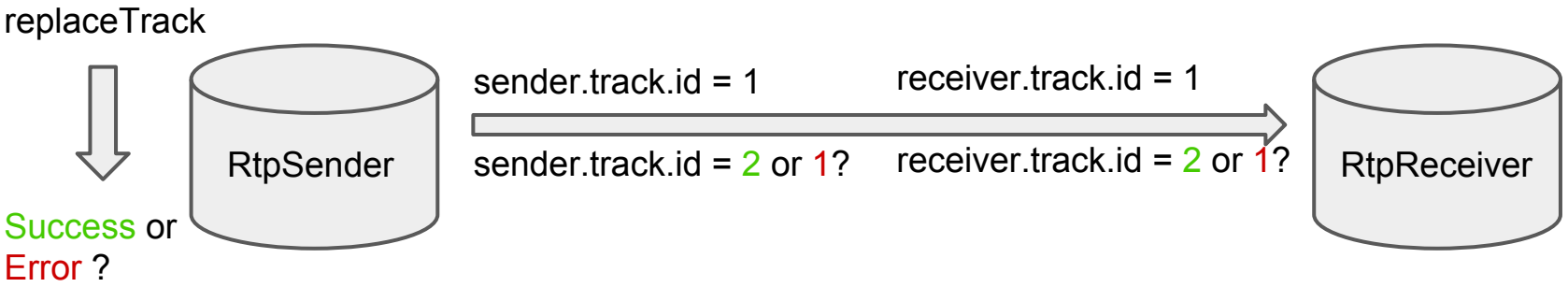


Things like dimensions and frame rate do not require negotiation. SDP is kept unchanged by marrying `msid` to the sender rather than its track (deviates from the current Firefox implementation).

Why we're here: What if renegotiation is needed?

```
sender.replaceTrack(newTrack).then(() => log("Success!"), e => log(e));
```

POLA



Rare: raw vs. pre-encoded video and different audio channels require negotiation.

What is least astonishing? `NegotiationNeededError` or `"negotiationneeded"` event?

NegotiationNeededError

Pros:

- Inherently less complex.

Cons:

- Fails where it has the information to succeed.
- People aren’t going to go the “extra mile” (e.g. slide 2), which effectively means giving up on rare video encoding and audio channel differences.

Rationale:

- replaceTrack is a narrow low-level API driven by avoiding renegotiation.
- We should make APIs below the signaling level to reduce complexity.

“negotiationneeded” Event

Pros:

- Potential to “just work” (iff pc.onnegotiationneeded is configured correctly)

Cons:

- Inherently more complex
- If pc.onnegotiationneeded isn’t configured, we never resolve and without clue.
- Need to figure out detecting negotiation failure (arguably not this PRs problem).

Rationale:

- replaceTrack is a high-level abstraction that may even survive SDP some day.
- We should make APIs above the signaling level that hide SDP.

Discuss