Galene
A videoconference server for the masses

Juliusz Chroboczek
IRIF, University of Paris Cité

25 May 2023

Galene

Galene is a videoconferencing system that was written during the first French lockdown.

- Optimised for teaching and conferences: (few senders $\longrightarrow$ many receivers);
- client: no installation (runs in browser);
- server: very easy to install;
- moderate server resources.

Current status:

- server: state of the art (with some minor exceptions);
- client:
  - fully-featured;
  - mediocre UI.

What’s so difficult about videoconferencing?

Videoconferencing is easy: *AT&T Picturephone* (1970).

Failed commercially: too expensive:

- dedicated hardware,
- dedicated network infrastructure,

and therefore no *network effect*.

Videoconferencing for the masses

Galene is extremely cheap:

`galene.org` costs 6€/month (plus tax),
0.10€/user/year.

We achieve this by:

- using existing devices: users' computers and smartphones;
- using existing infrastructure: the Internet.

Working with the limitations of existing infrastructure is hard.

User devices are broken

Most of Galene’s code runs on the users’ device:

- a server needs to be paid for;
- user devices have already been paid for.

But the user’s device is broken:
the “modern OS” dates from 1969:

- no application sandboxing;
- users have been trained to not install applications.

Difficult to publish code:

- Apple store: $$$;
- Google Play store: arbitrary rules;
- Linux distributions: herding cats is easier.

Solution: Galene’s client is a web application.

Web applications: the browser is the OS

Galene's client is a web application:

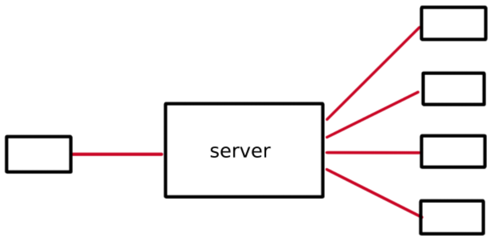
- written in JavaScript (nice);
- good sandboxing;
- users willing to view arbitrary web pages.

However:

- GUI using the DOM
  (Netscape C++ bindings from 1997);
- layout using CSS
  (like doing division in Roman numerals);
- deal with bugs in browsers
  (need a Mac for testing);
- deal with the web page's lifecycle
  (still less confusing than an Android view).

Throughput adaptation

Galene is an SFU: it forwards data from sender to receiver without reencoding.



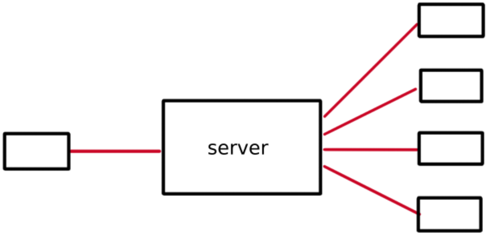
A non-adaptative SFU:

- ignores receivers' feedback,
- forwards the stream unconditionally.

Consequences: either

- video quality is poor (bad); or
- some receivers are overwhelmed (even worse).

Simple adaptative SFU



The first version of Galene was a simple adaptative SFU:

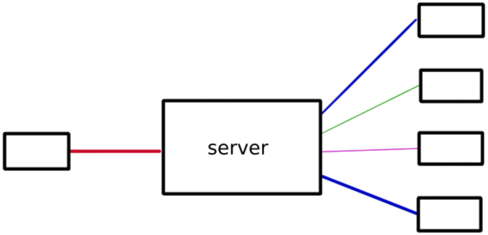
- listens to receivers' feedback;
- sender sends the highest quality that is acceptable to all receivers.

Consequences:

- high quality when all receivers are fast;
- slow receiver degrades quality for everyone.

Throughput adaptation: reencoding

Obvious solution: reencode the video at the server, in multiple qualities.

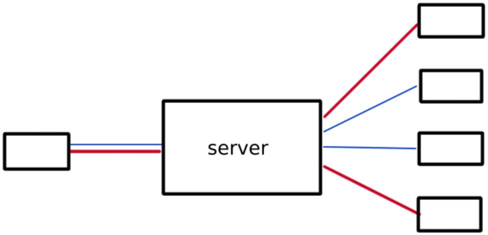


This is called an MCU:

- reencoding is CPU-intensive: expensive server;
- reencoding increases latency.

Simulcast

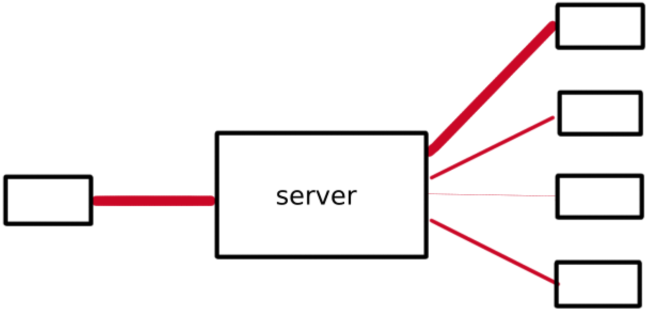
Simulcast is a simple technique for rate adaptation.
Requires cooperation from the sender.



- The sender sends multiple streams;
- the server selects a stream to send to each client.

Throughput adaptation: SVC

Scalable stream can be decimated by the server.



Two variants (can be used simultaneously):

- temporal scalability
  decimation reduces the frame rate;
- spatial/quality scalability
  decimation reduces the resolution or the quality.

Throughput adaptation in Galene

Galene implements both simulcasting and SVC.

Depends on the codec:

- VP8: simulcasting + temporal SVC;
- VP9: temporal and spacial SVC;
- H.264: simulcasting only.

Usually, the best stream is chosen by the server based on receiver feedback.

Tweak it manually: choose *Receive: Low* in the side menu to select the lowest-quality stream.

Recent developments

Since October 2024, Galene’s development is funded by NLnet.

- Security review: keeps the developer humble;
- background blur: no need to clean your room;
- speech-to-text: automatic subtitling;
- SIP gateway: back to the future.

(And also a lot of boring but necessary stuff, a client library for Go, a management interface and client, etc.)

Background blur

Background blur: essential for some users.

1. separate the foreground from the background;
2. blur the background;
3. composite the two.

Three techniques for step 1:

- green-screen (traditional);
- depth reconstruction;
- object recognition.

Background blur

We do object recognition using Google's selfie segmentation library:

- modified to not contact Google's servers;
- runs in a separate thread (the UI remains responsive);
- when backlogged, we drop frames, no lag.

We then perform background blur entirely on the GPU: no unblurred data ever leaves the local host.

(Not the obvious compositing algorithm.)

Speech-to-text

Galene-stt is a speech-to-text client for Galene.

- can do automatic subtitling;
- can generate a transcript.

Uses a self-hosted version of OpenAI’s Whisper:

- self-hosted: doesn’t contact OpenAI’s servers;
- requires a GPU for real-time usage.

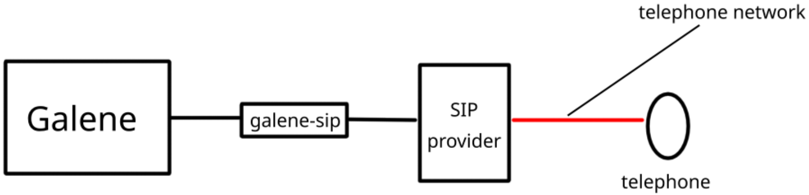
(Merci, J.-B.!)

SIP gateway

SIP is a protocol for VoIP (Internet telephony):

- designed in 1992, and it shows;
- interoperable, federated, standard (like e-mail);
- sometimes it even works.

Galene-SIP: gateway Galene $\longleftrightarrow$ SIP.



Join a Galene conference by making a phone call.

Conclusion

Galene is a videoconferencing system for the masses:

- easy to install;
- works over the public Internet;
- with existing user devices;
- the UI needs more work.

Working well over the Internet is hard:

- video: simulcast and SVC;
- audio: FEC, lipsync,
- network: NAT traversal,
- …

It is easy to build tools that work with Galene:

- galene-stt, galene-sip, galene-irc?