
# OUTILS LIBRES, SOBRES ET SOUVERAINS POUR LES COURS À DISTANCE EN DIRECT

Hugo RAGUET *[1]

[1]INSA Centre-Val de Loire

21 mai 2021

## RÉSUMÉ

Beaucoup de cours à distance forcés pendant la pandémie de COVID-19 se sont construits en essayant d'imiter à distance l'acte pédagogique qu'on aurait fait en présence. Dans cet article, nous ne discutons pas du bien-fondé de cette approche, mais des outils utilisés pour la mettre en œuvre. Nous démontrons qu'au contraire des usages massivement observés jusqu'à présent, il est possible de remplacer des outils très consommateurs de technologie et d'énergie, et gérés par des entreprises privées et souvent étrangères, par des outils aux besoins raisonnables, et partant, qui peuvent être administrés de façon souveraine.

## 1 INTRODUCTION

### 1.1 Contexte et constats

En ce premier quart de XXIe siècle, les outils numériques sont devenus indispensables au quotidien de la plupart des habitants des pays développés. Les quelques mises en gardes publiques sont peu relayées dans les institutions démocratiques et dans les médias, et les notions de sobriété et de souveraineté numériques sont généralement reléguées au second plan.

Lorsque l'État Français a forcé l'enseignement supérieur à s'organiser à distance comme mesure sanitaire, il n'a pas accompagné ses agents sur les approches pédagogiques possibles ni sur les moyens de les mettre en œuvre.

On le sait, les actes pédagogiques en présence ne peuvent pas être reproduits tels quels en ligne, même dans les conditions les plus favorables ; mais en l'absence de réelle formation et de temps pour s'adapter à une pédagogie nouvelle, c'est sans surprise que la plupart des enseignants ont cherché à remplacer le temps d'interaction en direct avec les étudiants qui a lieu en temps normal en personne dans des salles de classes dédiées, par des séances de vidéo-conférences en ligne.

Nous ne cherchons pas à discuter ici du bien-fondé de cette approche, mais des outils utilisés pour la mettre en œuvre.

*Enseignant-chercheur en informatique, hugo.raguet@insa-cvl.fr

Transmettre du contenu vocal et visuel à plusieurs dizaines de participants, en direct et de façon à ce que les auditeurs puissent réagir à leur tour, est loin d'être anodin. En l'absence d'investissement institutionnel adéquat, seules quelques grandes entreprises multinationales se sont avérées avoir les capacités physiques et la réactivité technique pour proposer des outils ergonomiques et fiables.

En seulement quelques mois, l'enseignement supérieur français est réputé ne plus pouvoir fonctionner sans le concours de l'entreprise Zoom, ou quelques uns de ses concurrents, comme Microsoft (qui développe Teams). Cela pose des problèmes évidents de contrôle des données et de souveraineté numérique.

### 1.2 Contribution

Pour éviter ces écueils, nous cherchons à éviter l'usage de la vidéo-conférence pour faire des cours à distance en direct. Notre contribution principale est une preuve de concept logicielle et matérielle, qui permet d'imiter les supports visuels de certains actes pédagogiques faits habituellement en présence, en n'utilisant que des outils libres et utilisant le minimum de ressources technologiques et réseau.

En conséquences, il devient possible d'utiliser des serveurs modestes pour la faire fonctionner ; même un petit établissement peut l'héberger lui-même. Puisque tous les outils sont en source libre [1], l'établissement contrôle toute la chaîne des données, sans avoir à payer un centime ou être tributaire d'intérêts privés et étrangers.

En second lieu, nos outils sont plus vertueux en termes de consommation d'énergie et de matériel ; d'accès à la connaissance pour les étudiants ayant peu de moyens ; de collecte de données puisque n'en collectant aucune ; et de partage de la connaissance puisque favorisant le logiciel libre.

Enfin, le support informatique utilisé étant plus adapté à l'information ciblée que ne l'est l'usage générique d'une vidéo, la qualité des supports visuels ciblés s'en retrouvent largement améliorée.

## 2 LES OUTILS PROPOSÉS

Notre preuve de concept se décline à ce jour en trois outils de communication visuelle en direct : un outil d'affichage et d'annotation en direct de diapositives, un outil d'affichage de texte formaté et d'équations mathématiques au fil de l'écriture, et un outil d'affichage de terminal virtuel au fil de son utilisation.

La partie client des ces outils reposent essentiellement sur des bibliothèques JavaScript libres et des normes du Web récentes, aussi ils ne nécessitent de la part des étudiants que l'accès à un navigateur Web moderne.

Cependant, avant de les détailler, il faut souligner que l'utilisation de supports visuels en direct est bien pauvre sans une interaction vocale pour les commenter. Nous commençons donc par présenter la solution de discussion vocale qui nous semble aujourd'hui le mieux les accompagner.

### 2.1 Préliminaire : voix sur IP avec Mumble

Pour correspondre à notre démarche, l'outil doit vérifier plusieurs critères. Premièrement, il doit permettre la communication vocale de la part de chaque participant et à destination de chaque participant, afin de permettre en particulier aux étudiants d'interagir et d'entendre les contributions de leurs condisciples.

Deuxièmement, il doit être le plus léger possible en terme de technologie et de besoin réseau ; en particulier, il doit être spécialisé dans le transport de la voix, et les considérations de vidéo doivent être ignorées.

Troisièmement, le logiciel correspondant doit être en code source libre.

Le logiciel Mumble (Mum) répond parfaitement à ce besoin. Plus précisément, la solution est constituée du logiciel client Mumble et du logiciel serveur Murmur, qui peut servir des centaines de clients avec du matériel et une maintenance modestes. [2]

Au delà des indispensables précités, le logiciel offre des fonctionnalités supplémentaires appréciables. En particulier une discussion textuelle pour ceux qui auraient des problèmes audio, des serveurs protégés par mot de passe, et de multiples salons de discussion virtuels pour lesquels il est possible de définir des règles de communication sophistiquées (quels utilisateurs peuvent parler, et quels utilisateurs peuvent les entendre). Par exemple, il est possible de faire interagir les étudiants par groupes indépendants les uns des autres, tout en donnant la possibilité à l'enseignant de se faire entendre par tout le monde.

Strictement parlant, le logiciel client n'est pas absolument nécessaire et peut être remplacé par une version en ligne, compatible avec tout navigateur Web moderne. Cependant, pour des raisons de performance il est

1. https://gitlab.com/1a7r0ch3/remote-lecture
2. https://wiki.mumble.info/wiki/Murmurguide

préférable d'installer le logiciel client sur le système des utilisateurs, ce qui ne pose aucun problème éthique car le code source est totalement libre. [3]

En attendant que nos institutions se décident à mettre sur pied un serveur Mumble, on peut déjà essayer l'outil grâce au serveur éthique de l'association Framasoft (Fra). C'est ce serveur que nous avons utilisé lors de nos cours à distance forcés depuis le mois d'octobre 2020 jusqu'à ce jour.

### 2.2 Affichage et annotation de diapositives

Le diaporama est beaucoup utilisé comme support de cours magistral. Son contenu est déterminé à l'avance, c'est donc l'outil le plus simple à mettre en œuvre en ligne : chaque diapositive est servie comme une image, affichée en plein écran. L'enseignant peut se contenter de faire défiler les diapositives à l'aide des touches directionnelles de son clavier, à mesure qu'il les commente.

Ce mécanisme est facilement mis en œuvre à partir d'un simple serveur HTTP dans l'environnement Node.js (Nod). Pour synchroniser l'affichage des diapositives sur tous les ordinateurs clients, nous utilisons socket.io (Soc). L'enseignant interagit directement avec le serveur par connexion SSH.

Si on se contente de servir la diapositive suivante au moment où celle-ci est commandée par l'enseignant, transmettre l'image à plusieurs dizaines de clients peut prendre du temps. À l'inverse, le réseau n'est pas du tout sollicité entre les changements. Aussi, pour éviter des délais parasites et lisser l'utilisation du réseau, nous exploitons les capacités de mise en cache des navigateurs Web, en commençant à envoyer les données des diapositives suivantes sans attendre que l'enseignant les commande. Passée la diapositive d'introduction, qui peut mettre plusieurs secondes à s'afficher, toutes les autres transitions sont instantanées pour tous les clients.

Il devient alors possible de servir de façon satisfaisante des diapositives de très bonne qualité à plusieurs dizaines voire centaines d'étudiants, même sur un réseau de dimension modeste.

Cette fonctionnalité est déjà intéressante, mais il manque un mécanisme qui permette de focaliser l'attention de l'auditoire sur des éléments spécifiques, voire de les annoter en direct. Aussi ajoutons-nous un outil de dessin en ligne. Nous adaptons à nos besoins et étendons des fonctionnalités du tableau en ligne WBO (Lojkine), mises en œuvre en JavaScript et reposant encore sur socket.io dans Node.js pour la synchronisation, en code source simple et libre. [4]

Cela comprend notamment un curseur visible par l'auditoire (qui peut faire office de pointeur laser virtuel), un outil de tracé à main levé, ou de lignes droites, rectangles ou ellipses, un outil d'insertion de texte rudimentaire, un outil pour effacer, et bien sûr la sélection de la couleur et de la transparence des tracés (cette dernière permettant par exemple de surligner). Notons que le dessin est indépendant d'une diapositive à l'autre, et qu'il est mis en mémoire, ce qui permet de le récupérer lorsque l'on revient sur une diapositive précédemment annotée.

Tous ces outils sont contrôlés par le dispositif de pointage de l'ordinateur. Les utilisateurs de stylet numériques peuvent déjà se servir de cet outil comme d'une série de tableaux de dimensions réduites, par exemple en utilisant une dizaine de diapositives vierges. Pour les autres, la souris ne permettant pas de reproduire la vitesse et la finesse de l'écriture manuelle, nous proposons un outil de tableau affichant du texte formaté.

### 2.3 Tableau de texte formaté en ligne

Même dans les études supérieures, les étudiants apprennent beaucoup par imitation, et certains actes pédagogiques nécessitent de transmettre non seulement un catalogue de connaissances et de techniques, mais encore le cheminement mental, tant rationnel qu'émotionnel, accompagné de la gestuelle physique, qui permettent de les mettre en œuvre. C'est particulièrement vrai pour les séances dites « de travaux dirigés ».

Il n'existe à l'heure actuelle aucune technologie qui puisse transmettre la gestuelle à distance, mais nous pouvons tenter au moins de développer, modifier et corriger le support visuel en même temps que le commentaire vocal, comme nous le ferions au tableau dans une salle de classe.

C'est l'objet de cet outil, qui consiste essentiellement à servir une simple page HTML, mise à jour à mesure que l'enseignant la rédige. Nous utilisons encore socket.io dans Node.js pour la synchronisation ; l'enseignant, connecté par SSH au serveur, doit utiliser un éditeur de texte qui écrit chaque modification sur le fichier à la volée.

On peut utiliser des simples balises HTML de style typographique (gras, italique, souligné, gros titre), mais il faut éviter d'avoir à gérer des balises de mise en page complexes en direct. En particulier, le tableau

3. https://github.com/mumble-voip/mumble
4. https://github.com/lovasoa/whitebophir

est organisé en deux colonnes, un fichier pour chacune d'elles, et l'enseignant est encouragé à utiliser des zones de texte dites « préformattées », qui conservent en particulier les espaces et les sauts de lignes.

Tout cela est déjà possible, de façon presque aussi sobre et peut-être plus ergonomique, à l'aide de nombreux outils de traitement de texte collaboratif existant, comme par exemple Etherpad (Eth). Mais ce qui distingue notre outil est la possibilité d'écrire des mathématiques en temps réel, ainsi que du code source avec coloration syntaxique.

Pour ces dernières fonctionnalités, il faut que la syntaxe dans le document HTML soit intuitive et légère, et que le moteur qui traduit la mise en page soit tout aussi efficace. Pour les mathématiques, nous utilisons le méconnu mais excellent module JavaScript jqMath (Barton) ; pour la coloration syntaxique, nous utilisons le célèbre module JavaScript Prism (Verou). Fait important, ces moteurs sont appelés du côté du client, ce qui malheureusement multiplie le besoin de calcul par le nombre de clients, mais permet d'alléger la charge de calcul du serveur à chaque modification du tableau.

Le résultat est illustré sur la figure 1. Signalons que l'utilisation de jqMath en direct nécessite un certain entrainement, en particulier pour écrire des symboles qui n'appartiennent pas à l'alphabet latin. Pour nos expériences, nous utilisons l'éditeur de texte Vim, muni d'un ensembles de raccourcis claviers personnalisés et sophistiqués.

### 2.4 Diffusion de terminal virtuel

Ce troisième et dernier outil s'adresse plus spécifiquement aux enseignements comprenant une forte composante de programmation informatique. Il s'agit de diffuser en direct un terminal virtuel sur lequel l'enseignant travaille.

Encore une fois la synchronisation utilise socket.io dans Node.js ; l'enseignant utilise un terminal sur le serveur via une connexion SSH. Le flux de données brutes du terminal est capturée par le logiciel script (inclus dans la suite util-linux), puis envoyé au client qui affiche le terminal dans le navigateur grâce au puissant module JavaScript Xterm.js (Xte).

## 3 CONCLUSION

Notre preuve de concept est encourageante, chacun de ces outils ayant rempli leurs promesses de simplicité et d'efficacité lors de nos essais préliminaires.

Nous pensons que ces outils et les avantages qu'ils représentent devraient d'ores et déjà nous permettre de nous passer des solutions de vidéo-conférence que l'ont met aujourd'hui en œuvre, au prix d'investissement matériels, financiers, mais aussi de l'indépendance numérique de l'enseignement supérieur.

Il reste néanmoins à les tester de façon plus exhaustive, à garantir leur fonctionnement et leur sécurité, à les adapter à un public plus large d'enseignants, et enfin éventuellement à rajouter des fonctionnalités, comme la possibilité pour les étudiants de pouvoir aussi contribuer directement sur ces supports visuels.

## RÉFÉRENCES

«Etherpad: highly customizable open source online editor providing collaborative editing in really real-time», URL https://etherpad.org/.

«Guide Mumble par Framasoft», URL https://docs.framasoft.org/fr/jitsimeet/mumble.html, ce serveur propose aussi le client en ligne https://web.mumble.framatalk.org/.

«Mumble: Open source, low latency, high quality voice chat», URL https://www.mumble.info/, ne pas confondre avec les serveurs d'hébergement https://www.mumble.com/ développée autour.

«Node.js: asynchronous event-driven javascript runtime», URL https://nodejs.org/.

«socket.io: real-time, bidirectional and event-based communication.», URL https://socket.io/.

«Xterm.js: bring fully-featured terminals in the browser», URL https://github.com/xtermjs/xterm.js.

Barton, D. «jqMath: Put math on the Web», URL https://mathscribe.com/author/jqmath.html.

Lojkine, O. «Wbo — collaborative whiteboard», URL https://wbo.ophir.dev/.

Verou, L. «Prism: lightweight, extensible and modern syntax highlighter», URL https://prismjs.com/.

**Exercice 30**

Suite de Fibonacci : $u_0 = u_1 = 1$ et $\forall n \geq 2$, $u_n = u_{n-1} + u_{n-2}$.

(a) Il s'agit d'une *suite récurrente linéaire d'ordre deux*, de polynôme caractéristique $X^2 - X - 1$, dont les racines sont $r_1 = \frac{1-\sqrt{5}}{2}$ et $r_2 = \frac{1+\sqrt{5}}{2} = \varphi$ (le nombre d'or). Or $|r_1| < 1$, donc ${r_1}^n \underset{n \to +\infty}{\to} 0$, et on déduit $u_n = \Theta(\varphi^n)$.

(b)

```
Algorithme Fibonacci : n → u selon
    Si n ≤ 1 alors u ← 1
    sinon u ← Fibonacci(n − 1) + Fibonacci(n − 2) .
.
```

Le nombre d'opérations autres que les appels récursifs est constant, donc la complexité temporelle de l'algorithme est proportionnelle au nombre d'appels récursifs. En notant $c_n$ le nombre d'appels nécessaires pour `Fibonacci(n)`, on a alors $c_0 = c_1 = 1$, et pour tout $n \geq 2$, $c_n = 1 + c_{n-1} + c_{n-2}$.
On peut poser $C_n = c_n + 1$, pour obtenir $C_0 = C_1 = 2$, et pour tout $n \geq 2$, $C_n = C_{n-1} + C_{n-2}$, c'est-à-dire pour tout $n \in \mathbb{N}$, $C_n = 2u_n$.
On en déduit $c_n = \Theta(\varphi^n)$. La complexité temporelle de l'algorithme est donc *exponentielle* en son entrée.

De plus, l'algorithme n'est pas récursif terminal, et la complexité spatiale est aussi proportionnelle au nombre d'appels, c'est-à-dire encore exponentielle.

Pour étudier expérimentalement ce qui précède, traduisons l'algorithme en langage C, qui permet les formulations récursives.

```c
int fibonacci(int n)
{
    if (n <= 1){
        return 1;
    }else{
        return fibonacci(n - 1) + fibonacci(n - 2);
    }
}
```

(c) Si l'on veut écrire une version itérative de cet algorithme,

```
<b>Exercice 30</b>
<br/>
<br/>

Suite de Fibonacci : $u_0 = u_1 = 1$ et
   $\forall n \geqslant 2, u_n = u_{n - 1} + u_{n - 2}$.
<br/>
<br/>

(a) Il s'agit d'une <i>suite récurrente linéaire d'ordre deux</i>,
de polynôme caractéristique $X^2 - X - 1$, dont les racines sont
$r_1 = \frac{1-\sqrt{5}}{2}$ et $r_2 = \frac{1+\sqrt{5}}{2} = \varphi$ (le nombre d'or).
Or $|r_1| < 1$, donc ${r_1}^n \underset{n \to +\infty}{\to} 0$, et on déduit
$u_n = \Theta(\varphi^n)$.
<br/>
<br/>

(b)
<pre><code class="lang-pseudo">
Algorithme Fibonacci : n → u selon
   Si n ⩽ 1 alors u ← 1
   sinon u ← Fibonacci(n − 1) + Fibonacci(n − 2) .
.
</code></pre>
<br/>

Pour étudier expérimentalement ce qui précède, traduisons
l'algorithme en langage C, qui permet les formulations récursives.

<pre><code class="lang-c">
int fibonacci(int n)
{
   if (n <= 1){
      return 1;
   }else{
      return fibonacci(n - 1) + fibonacci(n - 2);
   }
}
</code></pre>

<br/>

(c) Si l'on veut écrire une version itérative de cet algorithme,
```

FIGURE 1 – Capture d'écran du tableau en ligne et extraits des fichiers qui permettent de générer cet exemple.