Amélioration des pratiques collaboratives numériques des structures militantes par l'amélioration du logiciel libre de partage et de collaboration Nextcloud

Romain Lebrun Thauront

Du **6 septembre 2021** au **25 Février 2022**

Tuteur entreprise : Pierre-Yves Gosset
Tuteur technique : Tristan Roussillon
Tutrice humanités : Virginie Muller

Locaux Motiv'
10 bis rue Jangot
69007 Lyon
(+33 | 0)6 60 20 70 80
contact@framasoft.org

Bâtiment Claude Chappe
20 avenue Albert Einstein
69621 Villeurbanne CEDEX
(+33 | 0)4 72 43 60 60
tc-s@insa-lyon.fr

Remerciements

Je tiens à remercier l'ensemble de l'association Framasoft et l'ensemble des résidents des Locaux Motiv' du 10 bis rue Jangot pour leur accueil, et notamment l'équipe salariée de Framasoft, Angie, Anne-Marie, Aurore, Chocobozz, Josephk, Luc, Pouhiou, Pyg, Spf et Tcit. Pour votre approche humaine au monde du travail, votre franchise, votre soutien, votre bonne humeur et tout ce qui fait l'ambiance de travail à Framasoft ! Un merci tout particulier à Thomas CITHAREL pour ses réponses rapides et précises à mes questions alambiquées et à mon tuteur de stage, Pierre-Yves GOSSET, pour m'avoir proposé de prendre part à cette aventure et pour la relecture ortho-typographique de ce rapport.

Table des matières

Table des figures

1 Framasoft, une association d'éducation populaire aux enjeux du numérique

Framasoft est une association loi 1901 créée en 2004, qui a pour objet « l'éducation populaire aux enjeux du numérique et des communs culturels » [1]. Bien que majoritairement connue pour ses services web alternatifs et sa campagne « Dégooglisons Internet », ses actions sont très variées et vont du développement logiciel à la formation d'acteurs associatifs à des pratiques numériques saines.

1.1 Activités de l'association

1.1.1 Campagnes pour la sensibilisations aux enjeux numériques

Lancée fin 2014, la campagne de sensibilisation au capitalisme de surveillance et à ses alternatives nommée « Dégooglisons Internet » a mené, sur trois ans, à la mise en place de 37 services web afin de montrer que des alternatives aux GAFAM étaient possibles.

Cependant, malgré un afflux constant de visiteurs, maintenir cette diversité de services représentait une charge de travail trop importante pour l'association. Face à ce succès, au lieu de recruter et grossir, Framasoft a fait le choix de la décroissance via la fermeture progressive de plusieurs services avec la campagne « Déframasoftisons internet ». Ce choix politique contraire à la logique entrepreneuriale a été motivé par plusieurs raisons : Framasoft souhaite rester une association de petite taille avec au plus 10 salariés ; Framasoft souhaite voir plusieurs alternatives se créer plutôt que de devenir **la** grosse alternative ; Framasoft souhaite se concentrer sur d'autres objectifs que la proposition de services alternatifs.

D'autres grands projets ont succédés à Dégooglisons : initiée fin 2017, la campagne « Contributopia » a pour but d'amener un public plus large à contribuer aux communs culturels et notamment au logiciel libre, mais aussi d'apprendre comment intégrer ces contributions lorsque l'on mène un projet en sources ouvertes. Fin 2021, Framasoft lance 2 projets visant à faire de la capacitation numérique de certains acteurs ciblés avec du numérique responsable et éthique. Le premier, « Émancip'asso », a pour objectif de permettre aux associations qui souhaitent ne plus utiliser les outils de Google, Amazon, Facebook, Apple et Microsoft (GAFAM) de franchir le pas en formant des prestataires à l'accompagnement de telles démarches. Le second, « Framacloud », à pour but d'utiliser l'outil Nextcloud, un outil de collaboration et de partage de fichiers, comme prétexte et comme moyen de faire collaborer les acteurs de la justice sociale et du progrès social.

1.1.2 Développement de logiciels

Une des actions de Framasoft est le développement logiciel. Certains bénévoles et salariés participent au développement communautaire de logiciels libres utilisés par les services de Framasoft, comme le logiciel de liste de diffusion SYMPA ou

le logiciel d'écriture collaborative Etherpad. Mais Framasoft développe aussi ses propres logiciels, notamment PeerTube, un logiciel de partage de vidéos en ligne, Mobilizon, une alternative aux évènements Facebook et Yakforms, un logiciel de formulaires et d'enquêtes. Porter ces logiciels consiste à la fois à fournir un travail de développement important, à animer une communauté et à revoir et intégrer les contributions de la communauté au code source.

1.1.3 Mise en valeur de la culture libre

Framasoft a pour objet les communs culturels dans leur ensemble. Ainsi, Framasoft cherche à participer à des communs non-logiciels. Cela passe par l'utilisation de licences libres pour la publications des documents produits au sein de Framasoft, pour les livres édités par la maison d'édition Framabook, et pour les productions graphiques de l'association. Ces dernières sont obtenues par des prestations auprès d'artistes du mouvement de la culture libre comme David REVOY qui réalise la majorité des illustrations utilisées dans la communication et les logiciels de Framasoft.

1.1.4 Actions en réseau

La prise de position décroissante de Framasoft, afin de lui permettre d'explorer de nouveaux projets, pousse l'association à se séparer régulièrement de certains projets et à les transmettre à d'autres acteurs.

Cette partie de l'action de Framasoft est menée sur plusieurs fronts, l'un des plus importants étant le collectif CHATONS. Le Collectif des Hébergeurs Alternatifs, Transparents, Ouverts, Neutres et Solidaires est un groupe d'associations, de particuliers ou d'entreprises aux statuts divers proposant des services en ligne alternatifs. La fermeture de services de Framasoft a aussi pour but de laisser plus de place aux CHATONS sur la scène des services web alternatifs. Ce collectif a été fondé et est toujours coordonné par Framasoft.

Framasoft travaille aussi depuis longtemps en collaboration avec des acteurs du logiciel libre et du numérique éthique tel que l'APRIL ou la Quadrature du Net afin de faire connaître et défendre les valeurs communes avec ces partenaires. Néanmoins, Framasoft cherche à former de plus en plus de partenariats avec des structures menant des luttes différentes que celle du logiciel libre. Le but est de réaffirmer les intentions politiques (bien qu'apartisanes) de Framasoft, de travailler à une convergence des luttes, et d'apprendre en observant des structures différentes de la sienne.

1.2 Positionnements économiques et sociaux

1.2.1 Financements

En 2020, l'association Framasoft est financée à 93,5% par les dons [2]. L'association n'a donc pas de clients en tant que tels mais des bénéficiaires qui manifestent leur soutien par le don monétaire. Certains développements logiciels rentrent aussi

dans le cadre de dotations d'organismes privés ou publics. Par exemple, plusieurs fonctionnalités de PeerTube et Mobilizon ont été financées par la fondation NLnet comme la recherche de vidéos sur l'ensemble des instances PeerTube.

L'association n'est cependant pas contre les prestations de développement sous la condition que ces développements concernent les logiciels Mobilizon, PeerTube ou Yakforms et qu'ils ne soient pas en conflit avec la conception et les valeurs de ces logiciels.

Ce financement par les dons est à double tranchant : il permet une grande liberté d'action à Framasoft qui est libre de choisir comment utiliser ses fonds mais il crée une dépendance aux dons qui est un revenu potentiellement instable.

1.2.2 Concurrence et économie de marché

Framasoft étant une association loi 1901 dont les ressources reposent quasi-exclusivement sur les dons (environ 500 000€ par an), on peut la considérer comme une structure « hors marché ». La structure n'est donc pas en concurrence avec d'autres entreprises, car son but n'est pas d'occuper ou dominer un marché ou un secteur économique. Elle ne peut donc pas être placée sur une logique de concurrence avec les GAFAM, non seulement à cause de sa taille volontairement contrainte, mais aussi de ses objectifs intrinsèques (respect de la vie privée, notamment).

Par ailleurs, Framasoft ne compte pas vraiment d'équivalent, que ce soit en France, en Europe, voir dans le monde. En effet, l'association se trouve au croisement de différents domaines : éducation populaire, hébergement de services web (SaaS), et développement logiciel. Cela lui confère une place singulière, notamment dans l'écosystème du logiciel libre.

Cependant, les services proposés par Framasoft peuvent concurrencer d'autres services alternatifs. L'association est consciente de cette situation et cherche à limiter son impact afin de disséminer l'offre d'alternatives. Ceci par des mécanismes de limitations de nombres de comptes ou d'espace de stockage sur divers services Framasoft, et de la mise en valeur des alternatives concurrentes.

1.2.3 Les forces et faiblesses de l'association

Framasoft est une association aux projets aussi variés que ses membres, et en plein pivotement de son activité phare, de « propositions de services alternatifs » à « mener la capacitation numérique de structures militantes ». L'association est assez connue pour son offre de services en ligne mais sa nature associative et ses prises de positions politiques restent quand à elles assez méconnues. Il y a donc un risque d'incompréhension du public face à ce repositionnement qui se fait déjà sentir avec la fermeture de services. L'association mise donc sur ses compétences communicationnelles, sa transparence et ses prises de décisions collectives pour mener à bien ce projet.

Ces différents points de préoccupations sont synthétisés dans la matrice SWOT suivante (figure 1).

	Positif	Négatif
Interne	**Forces** Prise de décision démocratique et participative. Compétences techniques et communicationnelles fortes.	**Faiblesses** Coopération difficile entre les bénévoles et les salariés (pas les mêmes disponibilités ni les même attentes).
Externe	**Opportunités** Reconnaissance du public, bonne image médiatique. Réseau important au sein des millieux militants et des millieux de l'économie sociale et solidaire.	**Menaces** Méconnaissance du public sur les objectifs et la taille réelle de Framasoft. Notamment visible sur la campagne de fermeture de services. Dépendance aux dons. Réactions négatives de certains acteurs ou publics face à des positions politiques plus assumées.

FIGURE 1 – Matrice SWOT de l'association Framasoft en 2021

1.3 Organisation du travail dans une structure mêlant salariat et bénévolat

Framasoft est une association « loi 1901 » qui est composé de 37 membres, bénévoles et salariés. Il n'y a, dans les statuts, pas de diférences entre membre salarié ou bénévole. Elle est dirigé par une codirecion constitué de Pierre-Yves GOSSET et Pouhiou NOÉNAUTE. Le rôle de la codirection est de coordonner l'ensemble des actions de l'association. La coordination de certains domaines est déléguée à des comités spécifiques comme par exemple les Ressources Humaines (Voir annexe A). Les Comités sont exécutifs et ne font pas parti de la prise de décision.

Lorsque des décisions impactantes pour l'association sont à prendre, elles se font d'une manière qui se veut démocratique. L'ensemble de l'association est informée des tenants et aboutissants, et si le sujet fait débat on procéde à un vote par majorité. Ce vote peut se tenir en assemblé générale (AG) pour les sujets les plus importants (Voir annexe B).

S'il n'y a pas de différences entre les membres bénévoles et salariés pour ce qui concerne les prises de décisions, les deux groupes ont des attentes, des besoins, et du temps disponible différents et il faut concilier ces différences. Ainsi, il existe à la fois une volonté d'impliquer au plus possible l'ensemble de l'association et une volonté de laisser une certaine autonomie aux salariés. Par exemple, bien que la majorité des canaux de discussions instantanés et listes d'e-mail soit ouverts à tous, il existe des canaux réservés aux salariés pour la vie quotidienne au travail et certains aspects organisationnels qui ne relèvent que du salariat. L'autonomie des salariés passe aussi par l'autonomie de tous les membres qui sont libres de prendre et d'assumer des décisions qu'ils ne considèrent pas impactantes pour l'association.

2 Framacloud, créer de la collaboration entre structures militantes

Mon stage au sein de Framasoft s'inscrit dans le lancement d'un projet plus vaste évoqué dans la section 1.1.1, le projet « Framacloud ». Ce projet a pour but d'utiliser l'outil Nextcloud comme levier de la collaboration entre les acteurs de la justice sociale et du progrès social [1].

Nextcloud est un logiciel client-serveur de partage de fichiers et de collaboration en ligne et peut être vu comme une alternative libre à Google Workspace. Il s'agit d'un fork d'OwnCloud jouissant d'une communauté d'utilisateurs grandissante et d'un support commercial par l'entreprise Nextcloud GmbH qui dirige son développement.

Le projet Framacloud part du constat que la maîtrise de l'outil numérique, et notamment ces dernières années la capacité à collaborer en ligne, est assez déterminant pour l'impact d'une structure sur la société. Les associations, et plus encore les associations militant pour le progrès et la justice sociale, sont particulièrement fragiles face à cette problématique. Si l'usage de l'informatique est bien rentré dans le travail associatif, les associations souffrent souvent d'un manque de compétences, et d'un manque d'alternatives éthiques [3] (ce dernier point pouvant être crucial pour des structures militantes).

Les objectifs de Framacloud sont donc de mettre en avant une plateforme de collaboration pour les structures militantes, de l'adapter aux besoins associatifs et de pouvoir y fournir un accès à toutes les petites structures militantes en ayant besoin[4].

Pour pouvoir atteindre ces objectifs, Framasoft a misé sur Nextcloud. Malgré ces divers défauts, Nextcloud semble être la solution alternative la plus aboutie et la plus prometteuse. Le projet passera par des développements Nextcloud, mais aussi par l'augmentation de l'offre Nextcloud actuelle de Framasoft de 5000 comptes à 50 000 comptes.

Le lancement de Framacloud était prévu pour fin 2020, mais le contexte de pandémie des dernières années a retardé le projet. Ainsi le début de mon stage marque le lancement du projet. J'ai donc eu à réaliser, en plus d'un travail de développement, un travail exploratoire pour affiner notre compréhension des perspectives envisageables via ce logiciel. J'ai aussi été assez autonome dans mon travail de développement car j'étais le seul développeur sur ce sujet. J'ai néanmoins eu un suivi de projet, notamment via mon tuteur de stage Pierre-Yves GOSSET qui travaille aussi sur le projet Framacloud. J'ai aussi pu bénéficier des conseils techniques de mes collègues et notamment de Thomas CITHAREL, qui est un des principaux contributeurs du *plugin* de gestion d'agendas de Nextcloud.

1. Par ce terme d'acteurs, nous souhaitons englober un ensemble vaste d'organismes : des associations, des syndicats, des collectifs informels, des entreprises, ...

3 « Sorts » : Donner aux utilisateurs Nextcloud une vu sur leurs fichiers

L'objectif de ma mission de stage est d'améliorer l'utilisation qu'aurait un public militant et associatif du logiciel libre Nextcloud. Il s'est déroulé en trois parties, comme représenté sur le diagramme de Gantt du déroulé de mon stage en annexe C : une première partie de formation et de montée en compétences, une deuxième partie d'enquête et d'analyse de besoins, et une troisième partie de conception et réalisation d'une solution technique répondant à un des besoins identifiés.

Ce travail d'identification et de réponse à un besoin a donné lieu au développement de « Sorts », un *plugin* Nextcloud permettant aux utilisateur de mieux visualiser leur fichiers grâce à un explorateur de fichiers arborescent et des filtres conditionnels.

3.1 Prise en main du développement de Nextcloud et des processus de collaboration liés

Lors de cette première partie de stage mon objectif était d'acquérir suffisamment de compétences et de connaissances sur le développement du logiciel Nextcloud pour pouvoir choisir le développement logiciel sur lequel j'allais travailler lors de la troisième partie.

Cette montée en compétences fut un peu longue mais elle était nécessaire pour discerner la faisabilité et le temps de travail requis par les propositions d'améliorations des utilisateurs de Nextcloud que nous avons sondé lors de la deuxième partie du stage. De plus, la fin de cette montée en compétences était entrelacée avec la réalisation d'une enquête et ma participation au support de Yakforms (dont nous parlerons ci-après) qui ont induit du délai sur la fin de cette partie. (Voir diagramme de Gantt en annexe C)

3.1.1 Environnement de développement

Nextcloud étant un fork d'OwnCloud, son développement commence avec le développement d'OwnCloud en 2010. Ainsi sa *stack* logicielle d'origine suit le modèle classique des applications web LAMP : Un backend en PHP, un Frontend en JavaScript avec jQuery, distribué par un serveur Apache et soutenu par une base de données MySQL.

Cependant, le logiciel a su se moderniser : d'autres bases de données et serveurs web sont supportés, les développements actuels sont dans des versions récentes de PHP (PHP 7 et 8) et le frontend est progressivement réécrit en VueJS [5]. J'ai donc développé avec ces deux technologies, et mis en place le serveur de développement suivant afin de tester et visualiser mes développements en local :

— Serveur Apache (localhost)
— Base de données SQLite (localhost)

— PHP8 avec debuger Xdebug
— Nextcloud (Git-release, puis Nextcloud 23)

Sur le dernier mois de développement, nous avons aussi mis en place un serveur de test externe lié à un système d'intégration continue GitLab (voir partie 3.3)

Si j'avais déjà eu de premières expériences de développement web lors de mon cursus à l'INSA et au cours de mon stage de 4ème année à equensWorldline je n'avais jamais travaillé ni avec PHP, ni avec VueJS. J'ai donc commencé par apprendre les fondamentaux de ce langage et de ce framework. Puis, j'ai réalisé des premiers *plugins* factices et contribué au développement de Nextcloud en réglant des *good first issues*.

3.1.2 Fonctionnement général d'un *plugin* Nextcloud

Ma montée en compétences s'est faite via l'exploration de la documentation développeur, ce qui m'a permis non seulement d'apprendre le langage PHP mais aussi de comprendre le fonctionnement du système de *plugin* de Nextcloud.

Un *plugin* Nextcloud consiste en un dossier contenant l'ensemble du code backend et frontend de l'application, ainsi que ses dépendances et des informations d'identification. La figure 2 détaille les principaux dossiers d'un *plugin* Nextcloud. Le code PHP et les données d'identification du *plugin* sont découverts par Nextcloud via des mécanismes de réflexivité. Les URL définis dans le plugin sont notamment enregistré dans le routeur de Nextcloud pour charger le plugin lors de requêtes sur ces URL.

Un *plugin* Nextcloud a aussi accès à différentes APIs mises à disposition par Nextcloud et qui permettent au *plugin* de s'intégrer au logiciel, et d'accéder à des structures de données et des fonctionnalités de Nextcloud comme les fichiers, les utilisateurs, les droits d'accès ou les partages.

Pour le frontend, l'API prend la forme de fonctions JavaScript par défaut, de dépendance pré-paramétrées pour Nextcloud (par exemple un client REST ajax pré-paramétré avec l'URL de l'instance Nextcloud), et de *components* VueJS pour construire une interface utilisateur cohérente entre les *plugins* et le *core* de Nextcloud.

Pour le backend, l'API prend la forme de `namespace` PHP : `\OCA` est dédié au code des *plugins*, `\OCP` est dédié aux APIs publique que les *plugins* peuvent importer et appeler afin d'accéder aux fonctionnalités de Nextcloud, et `\OC` est réservé aux APIs privées (utilisé par `\OCP`)

Afin de comprendre le fonctionnement décrit ci-dessus et le fonctionnement des principales APIs Nextcloud, j'ai développé deux *plugins* factices. Le premier est le *plugin* du tutoriel de Nextcloud, permettant la gestion de prise de notes simples, stockées en base de données ; le deuxième est un *plugin* original, permettant de gérer des *todo lists* simples, stockées dans des fichiers texte.

Ces deux expérimentations m'ont permis de comprendre les fondamentaux du

```
plugin_name/
|-- COPYING
|-- Readme.md
|-- appinfo/
|   |-- info.xml            //Identification du plugin
|   `-- route.php           //URL pour lesquels charger le plugin
|                             (et fonctions PHP à appeler)
|-- lib/                    //Code PHP, chargé automatiquement lors
|   |                         d'une requête
|   |-- AppInfo
|   |   `-- Application.php //Point d'entrée du plugin lors de son
|   |                         chargement automatique
|   |-- Controller          //Point d'entrée des requêtes http
|   |                         définis dans routes.php
|   `-- ...                 // reste du code PHP visant à la
|                              complétion des requêtes
|-- node_module/            // Dépendances du frontend
|-- src/                    // Code du frontend
|-- test/                   // Test Units pour tester les
|                              fonctionnalités du backend
`-- vendor/                 // Dépendances du backend PHP
```

FIGURE 2 – Principaux dossiers et fichiers d'un *plugin* Nextcloud

backend des *plugins* Nextcloud et le fonctionnement des APIs les plus utilisées, mais aussi d'apprendre à utiliser la documentation de Nextcloud.

3.1.3 Processus de collaboration dans le cadre du développement Open-Source

Cette période de montée en compétences m'a également permis de m'habituer aux processus de travail dans lesquels s'est inscrit mon stage. Il s'agit des processus de Framasoft où j'ai été intégré à l'équipe de l'ensemble des 10 salariés mais aussi des processus de collaboration dans le cadre du développement d'un logiciel Open-Source, riche en interactions avec la communauté Nextcloud et les développeurs de Nextcloud GmbH.

Pour ce qui est de la participation au développement de Nextcloud, les développements menés par Nextcloud GmbH suivent un schéma classique de développement Open-Source : le versionnement se fait via Git et le code est hébergé sur la plateforme GitHub qui sert de suivit de projet et de *bug-tracker* et est une porte de communication avec la communauté. Cependant, toutes les interactions sur GitHub sont centrés sur le code et les aspects techniques. Le forum d'aide de Nextcloud est

donc là pour la communication plus générale entre Nextcloud GmbH et la communauté.

Ne sachant pas encore au début de mon stage si mon sujet technique consisterait en un *plugin* ou des développements au *core* de Nextcloud, cette montée en compétences avait aussi pour but de réaliser un première contribution au développement de Nextcloud. J'ai donc contribué à différents *plugins* officiels de Nextcloud en réglant des petits bugs ou en développant des petites fonctionnalités. J'ai par exemple rajouté un bouton permettant de vider les « corbeilles » des applications Calendar et Todo (voir figure 3 et merge request publique sur GitHub `https://github.com/Nextcloud/tasks/pull/1802`).

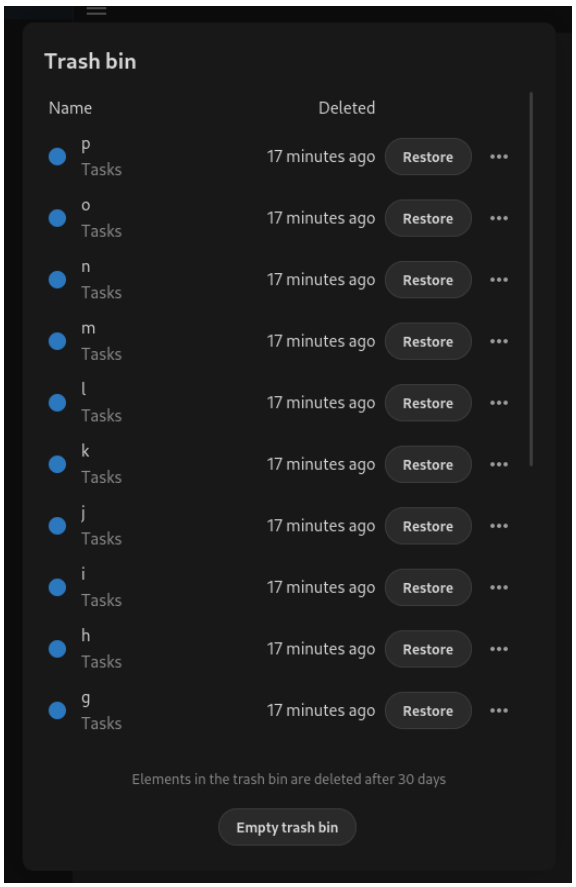


FIGURE 3 – Interface de la corbeille de l'application Todo, avec bouton de suppression de tous les items (*Empty trash bin*)

Cette première contribution m'a permis de mieux comprendre le fonctionnement d'un frontend en VueJS, notamment l'agencement en *components*, mais surtout d'appréhender le processus de contribution à un projet Open-Source :

1. Un *bug* ou une *feature* manquante est identifié

2. Création d'une *issue* sur le *repository* github correspondant
3. Un développeur souhaite régler cette *issue* et fait part de sa volonté dans les commentaires de l'issue
4. Fork du code source en question
5. Ouverture d'une nouvelle branche Git
6. Développement
7. Merge request citant l'issue réglée
8. Retour d'un des *maintainer* du *repository* en question et amélioration de la merge request jusqu'à ce qu'elle passe les tests automatiques (*pipeline*) et qu'elle soit approuvée
9. La merge request est intégrée au code upstream et l'*issue* est marquée comme résolue.

Lorsque nous avons ouvert le développement du *plugin* « Sorts » aux contributions internes à Framasoft, j'ai put aussi voir ce processus du côté *maintainer*, en relisant et acceptant les merge request proposée par mes collègues (voir partie 3.3).

3.2 Conception d'un *plugin* : de l'analyse du besoin à la solution technique

Au cœur du projet Framacloud, il y a la volonté de démocratiser le numérique collaboratif et de répondre aux besoins des collectifs militants pour le progrès et la justice sociale.

Nous avions donc une volonté forte d'identifier les besoins des utilisateurs ciblés avant de se lancer dans une réalisation technique. Ceci s'est concrètement traduit par un travail d'exploration de Nextcloud et la rédaction d'un court rapport d'étonnement qui fait place des frictions que j'ai pu avoir à l'utilisation du logiciel, mais surtout à la réalisation de l'enquête « Nextcloud et les structures du progrès social et de la justice sociale » dont les résultats seront publiés au court de l'année.

3.2.1 Études des limites de Nextcloud dans un usage associatif

Pour nous accompagner sur la création et la cohérence du formulaire d'enquête, et ainsi limiter le nombre de biais que nous aurions pu avoir à le concevoir et le diffuser par nous-mêmes, nous avons fait appel à deux prestataires externes : la designer Marie-Cécile Godwin et Stéphanie Lucien-Brun de La Fabrique à Liens, spécialiste du « numérique au service du pouvoir d'agir ».

Le but de cette enquête était de cibler les points de frictions qu'avaient rencontré des structures du progrès social et de la justice sociale utilisant déjà Nextcloud, ainsi que leurs attentes et leurs usages du logiciel. Ayant dorénavant une bonne vision des fonctionnalités de Nextcloud, j'ai proposé des premiers prototypes de sondage

qui ont nourri nos réflexions sur les questions à aborder pour traiter des différents cas d'usage du logiciel et sur lesquels Marie-Cécile Godwin a pu s'appuyer pour construire le formulaire d'enquête final.

Cette enquête a été diffusée fin octobre via les canaux de communication de Framasoft. J'ai ensuite été chargé de faire une synthèse des 180 premiers résultats recueillis sur une dizaine de jours d'enquête avec pour but de mettre en valeur les attentes les plus importantes du public ciblé.

Sur 174 résultats, 143 résultats correspondaient au public visé et ont été retenus dans mon analyse.

Afin d'extraire des tendances de ces résultats, j'ai rajouté manuellement des mots-clefs sur chaque réponse, caractérisant les préoccupations principales de la réponse. J'ai ensuite, à l'aide d'un script python, trié les mot-clefs par nombre d'occurrences et compilé dans un document, pour chaque mot-clef, toutes les réponses y référant. Enfin, à partir de ce document, j'ai rédigé une synthèse reprenant par mot-clef les frictions et attentes des répondants.

Les préoccupations les plus importantes ainsi relevées étaient les suivantes :

1. L'édition collaborative de documents (texte, tableur, ...) n'est pas assez performante
2. Nextcloud n'est pas assez performant (lenteur du logiciel)
3. L'application de visioconférence intégrée n'est pas assez performante par rapport aux alternatives
4. La synchronisation des fichiers avec des appareils est trop erratique
5. Les utilisateurs ont du mal à s'y retrouver parmi l'arborescence de fichiers de leur organisation.
6. La fonctionnalité de partage n'est ni intuitive ni ergonomique

Parmi ces problèmes, les deux premiers sont hors de notre champ d'action. Le premier concerne des logiciels externes (OnlyOffice ou Collabora) et participer à leur amélioration aurait nécessité une nouvelle période de formation de quelques mois. Le deuxième concerne l'optimisation en général de l'ensemble du logiciel, sur laquelle nous ne pouvons pas faire grand-chose sur une durée de stage limitée.

Les sujets 3 et 4 sont des sujets sur lesquels nous aurions eu plus de capacité d'action, mais pour lesquels les développements logiciels risquaient de ne pas rentrer dans le temps de stage restant.

Nous avons donc décidé de partir sur le quatrième sujet et d'aider les utilisateurs à mieux comprendre la hiérarchisation de leurs fichiers et des dossiers partagés par leur organisation.

Cette enquête a été cruciale pour le choix de mon sujet technique et pour guider sa réalisation, mais elle est aussi une ressource importante pour le reste du projet Framacloud. Il faut néanmoins garder à l'esprit qu'elle présente quelques biais,

notamment le fait que parmi ces 143 réponses, 50% des répondants déclarent travailler dans le numérique et sont donc susceptibles d'avoir une culture numérique non représentative du groupe d'utilisateurs ciblé.

De plus, en ce qui concerne ma synthèse, ma catégorisation manuelle et mon effort de synthétisation induit nécessairement un biais qui m'est propre, quand bien même j'ai essayé de rester autant que possible fidèle aux réponses originelles.

3.2.2 Prototypage de « Sorts »

Nous avons cherché comment améliorer la navigation des utilisateurs à travers leurs fichiers, tout en étant limité par plusieurs contraintes.

La première est qu'il restait trois mois de stage à consacrer au développement, parmi lesquels se trouvaient les congés de fin d'année, la rédaction du rapport de stage et la soutenance de stage.

La seconde est que l'application « Files » officielle est très *legacy*. La majorité de son code date du fork avec OwnCloud et n'a pas connu de développement majeur, Nextcloud GmbH à prévu de la réécrire prochainement mais ce n'est pas dans la feuille de route des deux prochaines versions. Proposer des modifications à l'application « Files » actuelle ou participer à sa réécriture ne nous paraissait donc pas envisageable.

Nous avons donc décidé de développer un *plugin* plutôt que de proposer des développements upstream. Le développement d'un *plugin* nous a effectivement permis d'avoir entièrement la main sur les fonctionnalitées que nous souhaitions développer et la certitude que mes développements soient publiés à la fin de mon stage.

Cependant, du point de vue du logiciel Nextcloud, avoir beaucoup de *plugins* avec peu de supports est une faiblesse. De plus, un *plugin* Nextcloud non-officiel risque d'être moins utilisé par les utilisateurs et notamment les associations. Ces dernières n'ont pas toujours l'infogérance de leur instance et pas ou peu de contrôle sur les applications qui y sont installés, et certains hébergeurs peuvent refuser d'installer des applications non-officielles.

Ainsi, nous ne visons pas tant, avec ce *plugin*, l'impact direct que l'impact indirect : en proposant de nouveaux concepts autours des interactions avec les fichiers, le *plugin* que j'ai développé fait office de *Proof Of Concept* qui, on l'espère, influencera la réécriture de « Files ».

Afin d'avoir une feuille de route pour le développement de ce nouveau *plugin* (appelé « Sorts »), j'ai définis avec mon tuteur de stage, Pierre-Yves GOSSET, l'ensemble des fonctionnalités que nous souhaitions implémenter afin de faciliter l'accès aux fichiers :

1. Proposer un nouvel explorateur de fichiers

   (a) Mettant en avant plus de metadatas, et notamment les tags que peuvent mettre les utilisateurs.
   (b) Permettant de « dérouler » l'arborescence de fichiers pour une meilleure compréhension du classement des fichiers et dossiers.
2. Proposer un système de filtres multiples
   (a) Permettant à l'utilisateur de combiner des conditions de recherche sur toutes les metadatas disponibles (taille, poids, tags, ...)
   (b) Permettant un accès rapide aux filtres les plus utilisés (par des filtres par défaut, un historique ou des favoris)
3. S'arrêter à la visualisation, et ne pas permettre la modification de fichiers
   (a) Permet de diminuer la « dangerosité » du *plugin* au cas où quelque chose se passe mal.
   (b) Permet de développer un produit fini dans le temps imparti (implémenter de la modification reviendrait à redévelopper l'application « Files »)
   (c) Les modifications peuvent se faire dans l'application « Files » officielle, en proposant un mécanisme pour ouvrir les fichiers et dossiers trouvés dans « Sorts » dans « Files »

À partir de ce scope j'ai créé un prototype interactif de l'interface utilisateur de « Sorts » grâce à l'application web « Penpot ».

La figure 4 présente la vue « explorateur de fichiers » du prototype : le panneau central consiste en une liste de fichiers avec le dossier « Projet » déroulé ; le symbole *external* après le nom de chaque fichier permet son ouverture dans `Files` ; dérouler un dossier se fait par un clic simple et entrer dans un dossier se fait via un double-clic. Sortir d'un dossier (remonter dans l'arborescence) se fait via le *breadcrumb* en haut de l'interface (symbole *Home* > Documents) ; les filtres sont accessibles via la barre de recherche à côté du *breadcrumb* et le panneau de navigation latérale permet d'accéder rapidement aux recherches favorites.

La figure 5 présente la vue de sélection des filtres et ses résultats : la partie supérieure de l'interface présente tous les critères de filtres combinables et la partie inférieure est similaire à l'explorateur de fichier, mais les fichiers présentés n'appartenant pas au même dossier, le chemin de chaque fichier est spécifié à côté de son nom. Les dossier y sont aussi déroulants pour permettre à l'utilisateur d'inspecter rapidement le contenu d'un dossier correspondant à la requête envoyée.

Cette étape de prototypage a permis de rapidement se rendre compte des éléments nécessaires à une bonne expérience utilisateur et a permis de rapidement récolter des retours sur la façon dont nous envisageons le *plugin*.

Nous nous en sommes notamment servi pour publier une note d'intention sur le forum officiel de Nextcloud, qui a donné lieu à des retours de la communauté Nextcloud mais aussi, bien qu'un peu tardivement, de développeurs de Nextcloud GmbH.

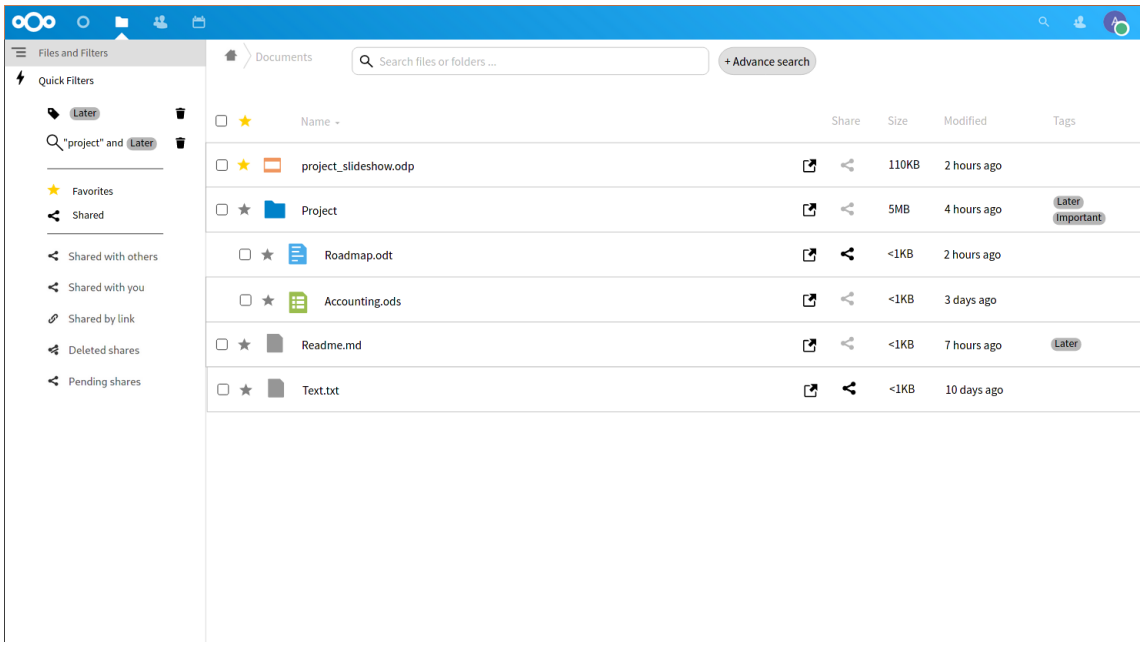


FIGURE 4 – Prototype interactif du *plugin* « Sorts », explorateur de fichiers

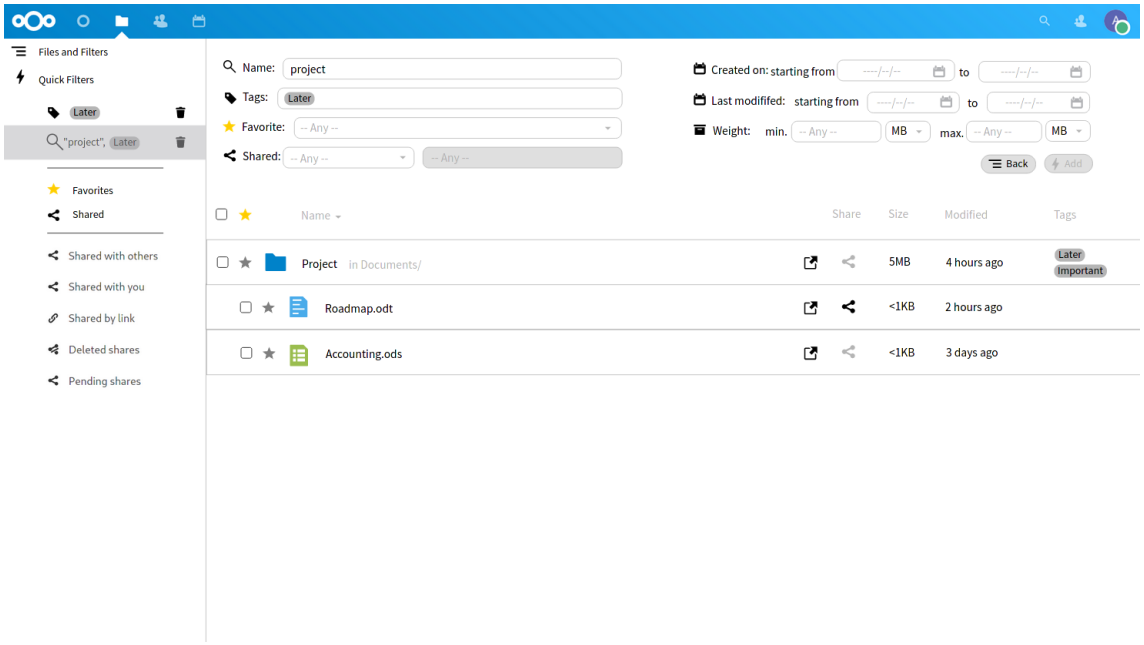


FIGURE 5 – Prototype interactif du *plugin* « Sorts », filtres combinés

Ces retours ont donné lieu à une réunion avec Nextcloud GmbH au sujet de « Sorts ». Cette réunion nous a permis d'établir un premier contact avec Nextcloud GmbH et d'envisager comment mes développements pourraient avoir la meilleure plus-value pour Nextcloud. Effectivement, la multiplication d'applications peu suivies n'est pas idéal pour l'écosystème du logiciel, mais, comme évoqué plus haut, les délais de mon stage ne me permettaient pas de pivoter mon sujet technique à un mois et demi de ma fin de stage. Nous avons donc gardé comme priorité le développement de Sorts.

Le prototype présenté ici s'est construit rapidement avec ces retours alors que redévelopper des fonctionnalités aurait été beaucoup plus coûteux en temps.

3.3 Processus de développement du plugin « Sorts »

Le développement de « Sorts » a commencé dès la fin du prototypage. Cependant le début de développement était plus une phase de mise en place du projet. Il s'agissait de re-modeler la base de code de l'application « tutoriel » de Nextcloud afin de partir sur un modèle d'application vierge, d'ouvrir un *repository* sur Framagit, l'instance GitLab de Framasoft, et de mettre en place un échéancier de développement.

J'ai donc fais une première décomposition du *plugin* en grande fonctionnalités, auxquelles j'ai attribué un temps de développement :

— Backend de l'explorateur de fichiers
— Frontend (UI) de l'explorateur de fichiers
— Backend de filtres simples (poids, taille, noms des fichiers, ...)
— Frontend (UI) de sélection des filtres
— Backend de filtres plus complexes (tags et information de partage)

Mais ces prévisions ce sont heurtées à la réalité du développement logiciel. J'ai notamment eu du mal à garder une vraie division en tâches sur les premières semaines de développement.

Partant d'une feuille blanche, le développement des premiers éléments du backend et du frontend ce sont retrouvés très interdépendants. Une fois une première version de l'explorateur de fichiers réalisé, il m'a été beaucoup plus simple de suivre des sous-développements de tâches définies avec le modèle classique d'une branche Git par *feature*. Cette amélioration se voit clairement sur l'historique Git du plugin qui voit apparaître des *commits* réguliers à partir de début janvier 2022.

Cette amélioration m'a permis de mettre un place un second échéancier plus juste des temps de développements, si on ne compte pas les refactorisations du code source, étape nécessaire avec ma compréhension des technologies utilisées qui grandissait au fil du développement.

Le temps de développement étant relativement court, je me suis concentré sur les aspects fonctionnels du plugin, et ai délaissé les considérations esthétiques de

l'interface utilisateur pour lesquelles j'ai moins de savoir faire et moins d'intérêt.

Ces aspects ont néanmoins pu être addressés fin janvier et début février avec la participation de JosephK et Thomas CITHAREL au développement de « Sorts ». Cette première ouverture du code de Sorts en interne nous a permis de mettre en place et de tester les processus de collaboration autour du *plugin*. J'ai pu relire, commenter et accepter des merge request de mes collègues sur les aspect esthétiques mais aussi sur des correctifs de bugs du *plugin*. Cette collaboration a nécessité que j'écrive la feuille de route du projet et priorise les tâches à réaliser pour une sortie d'une version 1 Beta, destinée à être publiée sur le magasin de *plugins* de Nextcloud.

Cette mise en place d'un environnement de développement à plusieurs a aussi permis la mise en place d'un système d'intégration continue et de *pipeline* GitLab semblable à ce qui se fait dans le développement logiciel professionnel.

L'intégration continue permet le déploiement des dernières modifications du plugin sur un serveur de développement grâce auquel nous avons pu visualiser et tester à plusieurs les derniers développements intégrés.

Le système de pipeline permettait quand à lui de tester le bon fonctionnement des nouvelles modifications avant leur intégration grâce à une chaîne de tests du code source, mais aussi d'uniformiser les conventions de codage du projet (*lint*).

3.4 Détails du fonctionnement de « Sorts »

Cette partie traite des principaux choix techniques réalisé au cours du développement du plugin « Sorts ». Sorts étant un logiciel libre et Open-Source, son code source est accessible en ligne à l'URL suivante : `https://framagit.org/framasoft/nextcloud/sorts`, et le lecteur peut s'y référer, s'il le souhaite, pour plus de détails sur le fonctionnement du plugin.

Le fonctionnement général d'un *plugin* Nextcloud ayant déjà été expliqué brièvement dans la section 3.1.2, j'aimerais préciser ci-dessous les grandes lignes du fonctionnement spécifique de « Sorts ».

3.4.1 Concepts généraux du Backend et du frontend

Le backend de « Sorts » implémente une API REST dont le but est de fournir au frontend de Sorts une liste de fichiers et de leur métadonnées respectives, via un objet JSON. Il n'implémente pas d'API pour des clients externes (*cross-origin*).

Les métadonnées sont toutes les informations concernant un fichier, qui ne sont pas son contenu, par exemple son nom, sa taille ou les tags qui lui sont associés. Les métadonnées qui sont affichés dans « Sorts » proviennent de différentes API publiques de Nextcloud [6] :

- `\OCP\Files`
  - Nom
  - Taille

    - Date de modification
    - Favoris
- `\OCP\SystemTags`
    - Tags
- `\OCP\Share`
    - Partagé avec
    - Partagé par
    - Partagé par lien

La gestion des différentes API est un des enjeux de « Sorts », à la fois parce que la récupération d'une liste de fichiers se traduit au mieux par trois requêtes à la base de donnée (une par API), et ensuite parce que les bases du système de recherche conditionnelle de Nextcloud font partie de `\OCP\Files` et l'intégration de paramètres portant sur les tags et les informations de partage nécessitent des compromis qui sont abordés dans la section 3.4.3

.

Le frontend de « Sorts » est une application VueJS. Il s'agit d'un assemblage de plusieurs composants VueJS, dont le détail est visible sur la figure 9, détaillée dans la partie 3.4.4.

Utiliser VueJS permet de prendre avantage de son système d'affichage *responsive* : lorsqu'une variable JavaScript lié à un élément de la page HTML de l'interface utilisateur est mis à jour, les éléments de l'interface utilisateurs liés à cette variable sont automatiquement rafraîchis (c'est notamment le cas de la liste des fichiers qui est stockée dans un tableau d'objet).

3.4.2 Design d'un explorateur de fichiers arborescent

Entre le prototype (figure 4), et le *plugin* (figure 6), il y a quelques différences. Nous avons choisi de laisser l'interface de sélection des filtres toujours visible, car étant la *feature* majeure du *plugin* nous la voulions toujours accessible. Nous avons aussi enlevé la possibilité de chercher par date de création, car, si l'attribut existe dans les objets PHP fournis par Nextcloud pour décrire un fichier, il possède toujours une valeur nulle.

Pour récupérer le contenu d'un dossier, une requête HTTP est faite au backend de « Sorts », qui réalise un premier appel à l'API « Files » pour récupérer la liste des fichiers contenus avec leur metadatas, puis cette liste est passée à deux autres API pour récupérer les tags et les informations relatives au partage du document. En bout de chaîne, ce processus résulte en trois requêtes à la base de données.

La question du nombre de requêtes à la base de données est un point de préoccupation majeur car il s'agit souvent du facteur le plus limitant sur de grosses instances avec plusieurs milliers de fichiers, d'autant plus qu'un explorateur de fichiers « déroulant » pourrait être amené à ouvrir beaucoup de dossiers d'un coup.

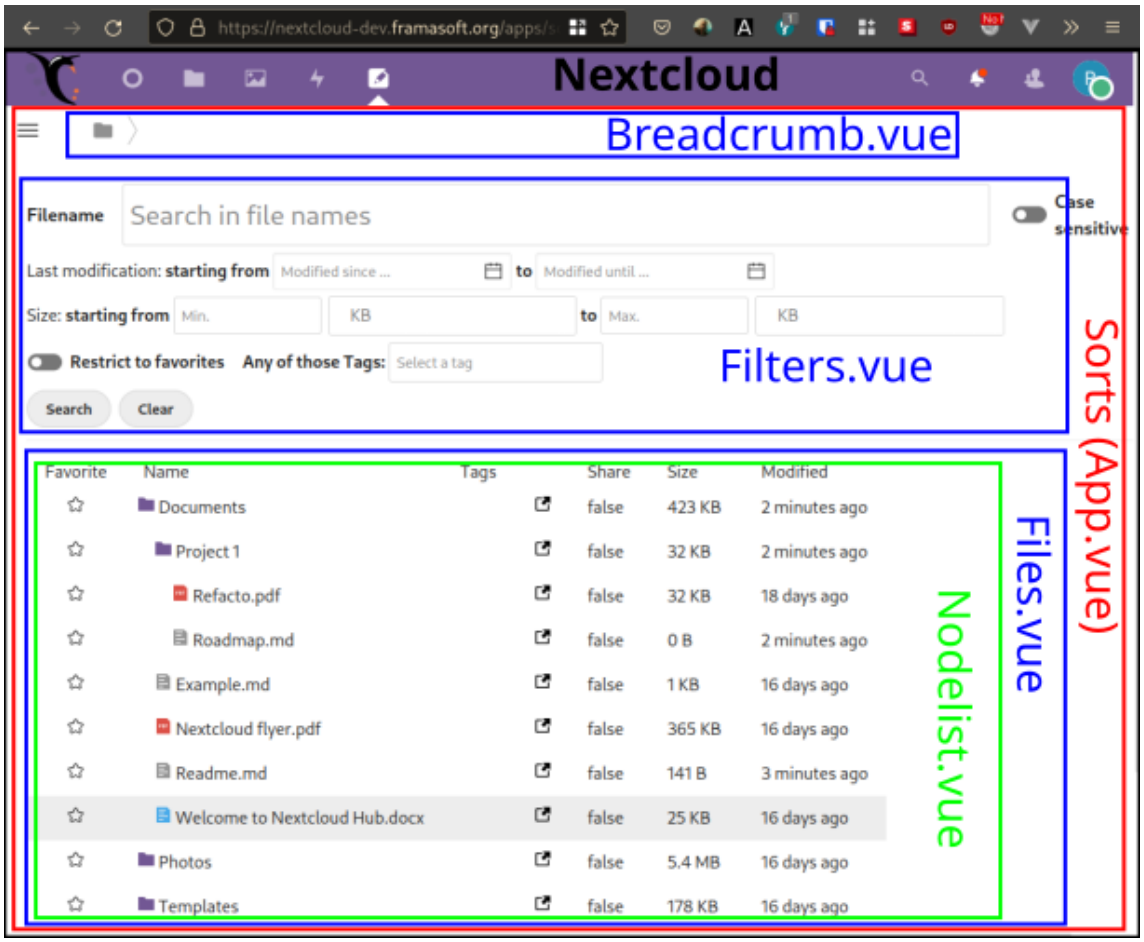


FIGURE 6 – Vue « explorateur de fichiers » de « Sorts »

La conséquence pourrait être de rendre le service inutilisable en submergeant le serveur de requêtes dans la base de données, ou de rendre l'interface utilisateur inutilisable en y chargeant beaucoup trop d'éléments d'un seul coup. Deux solutions nous paraissaient envisageables pour limiter ce problème : paginer les fichiers affichés et éviter les cas où plusieurs dossiers pouvaient être rechargés d'un seul coup.

La pagination nous paraissait adaptée à l'affichage des fichiers et dossiers répondant à un filtre (affichage de n résultats, avec un bouton pour en requêter plus), mais inadaptée à l'exploration de l'arborescence. Effectivement, tout l'avantage d'un explorateur de fichier arborescent est de pouvoir voir le contenu d'un dossier dans son « contexte », le contexte étant le contenu de ses dossiers parents.

J'ai donc pris la décision de ne pas paginer le contenu d'un dossier, mais d'éviter de recharger les dossiers de manière récursive. La conséquence est que lorsqu'un utilisateur remonte dans l'explorateur de fichiers le dossier parent est rechargé mais pas les dossiers enfants qui étaient déjà ouvert et qui peuvent donc ne plus être à jour. Deuxièmement, l'information des dossiers déjà ouverts est perdue au rechargement de la page. Ainsi, recharger une page ne rouvrira que le dossier à la racine de l'arborescence visionnée et pas tous les sous dossiers ouverts (le dossier actuellement à la racine de l'arborescence visualisée, ainsi que la metadata et l'ordre selon lesquels les fichiers sont affichés, sont stockés dans l'URL comme on peut voir aux figures 8 et 9, ils persistent donc au rechargement).

Mettre en place un mécanisme de rafraîchissement passif des fichiers visionnés lors de leur modification par une source externe est une des perspectives d'amélioration du *plugin*.

L'explorateur de fichiers arborescent posait aussi la question de comment afficher un arbre de profondeur variable (les fichiers, avec différents dossiers ouverts) avec VueJS. Les directives VueJS pour controler l'HTML ne contiennent que des conditions et des boucles *foreach* (*v-if* et *v-for*) mais la profondeur de l'arbre étant variable, son affichage nécessite un algorithme récursif ou des boucles *while*.

Pour résoudre ce problème, j'ai décidé d'aplanir l'arbre en un tableau en faisant appel à une fonction récursive `func` (`nodesList` étant le tableau aplani et `nodesTree` étant l'arbre de fichiers) :

```
computed: {
    /**
     * Return the plane array of nodes, with depth properties on each nodes
     *
     * @return {object|null}
     */
    nodesList() {
        if (this.nodesTree === null) {
            return null
        }
```

```
        const result = []
        const func = function(nodes, depth) {
            nodes.forEach((node) => {
                node.depth = depth
                result.push(node)
                if (Array.isArray(node.nodes)) {
                    func(node.nodes, depth + 1)
                }
            })
        }

        func(this.nodesTree, 0)
        return result
    },
}
```

Le tableau ainsi obtenu contient l'information de « profondeur » d'un fichier, qui est directement utilisée pour l'indentation des fichiers contenus dans un sous-dossier. Les nœuds d'un dossier ouvert sont aussi traités et insérés dans le tableau aplani directement après leur parent, ce qui permet de les garder sous leur parent respectif.

Ce tableau `nodesList` est déclaré comme une fonction car il s'agit d'une *computed property*. Il s'agit d'un mécanisme de VueJS : l'objet résultant d'une *computed property* est calculé à sa première utilisation et gardé en cache ; tant que les données avec lesquels la *computed property* ne change pas, le cache est retourné lors d'un accès plutôt que de recalculer la fonction.

De plus, VueJS recalcule la *computed property* et rafraîchi son affichage automatiquement lorsque les données sur lesquelles elle se base sont modifiés (dans notre cas, dès que l'arbre de fichiers `nodesTree` est modifié).

Sur la figure 6, nous pouvons voir que l'affichage des fichiers (et donc le code présenté ci-dessus) est géré par le composant `Nodelist.vue`. La raison de sa gestion dans un sous-composant plutôt que dans `Files.vue` est que l'affichage des fichiers est commun à l'explorateur de fichiers et à l'affichage des résultats des filtres. Le composant `Nodelist.vue` est donc réutilisé dans 2 contextes différents. La séparation en composant du *frontend* est abordé plus en détails dans la section 3.4.4.

3.4.3 Design des filtres conditionnels

Il existe deux modules de recherche disponibles via le framework PHP de Nextcloud. Le premier est celui des `SearchProvider` qui est utilisé dans la recherche globale de Nextcloud. Cette recherche globale permet de chercher une même chaîne de caractères parmi toutes les ressources présentes dans l'instance Nextcloud (que ce soit des fichiers, des rendez-vous, des mails ou des conversations). Chaque application voulant étendre cette fonctionnalité implémente l'interface `SearchProvider`

afin de retourner des résultats, et les `SearchProvider` liés aux fichiers ne cherche que dans les noms des fichiers ou dans le contenu des fichiers. Cette API, qui est la fonction de recherche la plus connue de Nextcloud ne correspond donc pas à des filtres conditionnels dont la plus-value se situe sur la capacité à filtrer d'autres caractéristiques et notamment le poids, la date de modification ou les tags.

J'ai donc utilisé dans la réalisation de « Sorts » la deuxième API de recherche : `\OCP\Files\Search`, qui implémente déjà un moteur de filtres sur les metadatas de `\OCP\Files` (voir section 3.4.1).

Ce moteur de filtres définit une interface `ISearchOperator`, qui contient les conditions de filtrage souhaitées, une classe `SearchOrder`, qui contient l'ordre et le nom de la metadata par lesquels trier les résultats, et une classe `SearchQuery` qui définit l'ensemble d'une requête : un `ISearchOperator`, un `SearchOrder` et des paramètres de pagination (le nombre de résultats maximum à retourner, et l'offset de ses résultats).

Deux classes implémentent `ISearchOperator` :

1. `SearchComparison` qui défini un filtre sur une metadata, composé de :
   - un *Field*, la metadata à comparer
   - un *Type*, la comparaison à appliquer :
     - égal (strict)
     - supérieur strict
     - supérieur ou égal
     - inférieur strict
     - inférieur ou égal
     - semblable (si la chaîne de caractère fournie est contenu dans le champ chercher)
     - semblable, mais avec la même case.
   - une *Value* : la valeur à comparer (un nombre, une chaîne de caractères ou un booléen)
2. `SearchBinaryOperator` qui défini une opération logique
   - AND qui s'effectue sur 2 `ISearchOperator`
   - OR qui s'effectue sur 2 `ISearchOperator`
   - NOT qui s'effectue sur un unique `SearchComparison`

Un ensemble de conditions est créé en combinant des `SearchComparison` dans des `SearchBinaryOperator`, dont l'empilement résulte en un `SearchBinaryOperator` contenant l'ensemble des conditions pour effectuer la recherche de fichiers souhaitée.

Ainsi, pour réaliser une recherche sur les fichiers d'un utilisateur, la classe `FilterManager` du backend de « Sorts » récupère une description JSON des filtres spécifiés par l'utilisateur, puis les `SearchComparison` correspondantes sont instanciées et combinées par des `SearchBinaryOperator` AND. Une `SearchQuery` est instanciée avec le `SearchBinaryOperator` et un `SearchOrder`, et est passée à la fonction `\OCP\Files\Folder::search()` du `Folder` à la racine des fichiers de l'utilisateur.

C'est cette fonction qui résoudra la requête en créant une requête SQL correspondante à l'aide d'un framework PHP de création de requêtes SQL.

La difficulté pour implémenter ces filtres sur les metadatas de l'API `Files` était de trouver et comprendre l'API `\OCP\Files\Search`, peu utilisée et peu documentée.

Son utilisation permet d'avoir confiance dans la justesse des requêtes SQL générées, d'avoir une seule requête SQL avec des jointures peu importe le nombre de critères choisis et a l'avantage de proposer de la pagination *out of the box*. Néanmoins, son problème est que ce système n'intègre pas les metadatas liées aux tags et aux informations de partage. Il faut donc passer par d'autres API pour filtrer ces metadatas et trouver un moyen de les combiner aux résultats existants.

En passant par l'API `SystemTags`, on peut récupérer la liste des fichiers étiquetés avec une certaine liste de tags et en passant par l'API `Share`, on peut récupérer la liste de tous les fichiers correspondant à une information de partage (partagé par moi, partagé avec moi par untel ou encore partagé via un lien public). Mais il s'agit de requêtes indépendantes. Ainsi, il y a besoin de 3 requêtes à la base de données pour filtrer sur l'ensemble des critères, et elles retourneront chacune autant ou plus de résultats que la requête combinée équivalente. Il faut ensuite recouper ces résultats d'une manière ou d'une autre, et comme il faut pour cela les réponses entières des trois requêtes, on perd l'avantage qu'on avait à paginer.

Sur une grande instance Nextcloud avec plusieurs milliers de fichiers, de telles requêtes peuvent s'avérer très lourdes et impacter le service. L'utilisation de « Sorts » dans sa version actuelle n'est donc pas forcément recommandé pour de telles instances. C'est un des problèmes le plus critiques du *plugin* et trois solutions sont envisageables pour y remédier :

1. Réécrire un système équivalent afin d'y intégrer la possibilité de rajouter les tables SQL contenant les informations sur les tags et les partages aux jointures.
2. Modifier l'API `\OCP\Files\Search` pour y intégrer des recherches sur les tags et les partages.
3. Ne pas faire de requêtes combinées pour ces deux metadatas mais une requête par API.

La première solution demande un temps de travail trop important pour le cadre du stage et la deuxième n'est pas envisageable car une telle modification d'API sera probablement refusé par Nextcloud GmbH. Effectivement, les modifications au *core* de Nextcloud ne sont pas acceptées sans nécessité forte et une longue période de tests afin d'être sûr de ne pas introduire de bugs dans les applications qui en dépendent.

« Sorts » effectue donc trois requêtes pour récupérer trois tableaux de fichiers qui *match* chacun les critères liés à une API, puis retourne au frontend l'intersection de ces trois tableaux.

Ce choix rend l'utilisation des filtres par tags et informations de partages non-viable pour des grosses instances Nextcloud et donc nous pousse à ne pas recommander l'utilisation de « Sorts » dans sa version actuelle. Il faudra dans le futur de l'application mitiger ce problème car les filtres par tags et informations de partage sont très attendues par les utilisateurs.

Une première amélioration possible est de procéder, dans le cadre d'un filtre portant sur les metadatas de l'API Files et d'au moins une autre API, de la manière suivante :

1. Récupérer tous les fichiers qui *match* les filtres sur les données de Files
2. Récupérer les informations sur les tags et le partage pour ces fichiers (comme lorsqu'on souhaite les afficher)
3. Parcourir ces fichiers avec une boucle, et filtrer en comparant leurs metadata de tags ou de partage avec un `if`

Si la requête sur l'API `Files` limite déjà fortement le nombre de fichiers possibles, alors les étapes 2 et 3 peuvent s'avérer bien moins lourdes qu'avec le système actuel car la quantité de données à récupérer et à traiter est moindre.

Une seconde amélioration serait de tout de même paginer les filtres combinant des informations de partage ou des tags avec des metadatas de `\OCP\Files`. Par exemple, pour retourner les 20 premiers résultats des fichiers modifiés au cours du dernier mois et ayant le tag « important », on pourrait procéder de la manière suivante :

1. Récupérer les 20 premiers fichiers modifiés au cours du dernier mois via `\OCP\Files`
2. Parmi ces 20 résultats, isoler ceux ayant le tag « important », soit par la méthode d'intersection des résultats soit par la méthode faisant appel à un parcours des résultats.
3. Si le nombre de résultats isolés est inférieur à 2O, requêter les 20 résultats suivant de l'API `\OCP\Files`, et retourner à l'étape 2
4. Sinon, on coupe les résultats isolés à 20 éléments, et on les renvoie avec l'index de pagination de l'API `\OCP\Files` auquel on s'était arrêté.
5. Les requêtes suivantes passent l'id du dernier élément reçu et, en tant qu'offset, l'index de pagination d'`\OCP\Files` plutôt que le numéro du « lot » de réponse qu'elles auraient normalement fournis.

Pour ce qui est du frontend, un *HTML form* permet de récupérer les critères de la recherche utilisateur (voir figure 7). Ce *form* fait usage de plusieurs composants de la librairie de composants VueJS de Nextcloud : les champs de sélection de la date, les champs à choix multiples, les *switch*, et la sélection de tags. La réutilisation de ces éléments permet un aspect plus unifié avec le reste de Nextcloud, mais aussi de ne pas « réinventer la roue » et d'être conforme aux préférences de choix de la langue des utilisateurs.

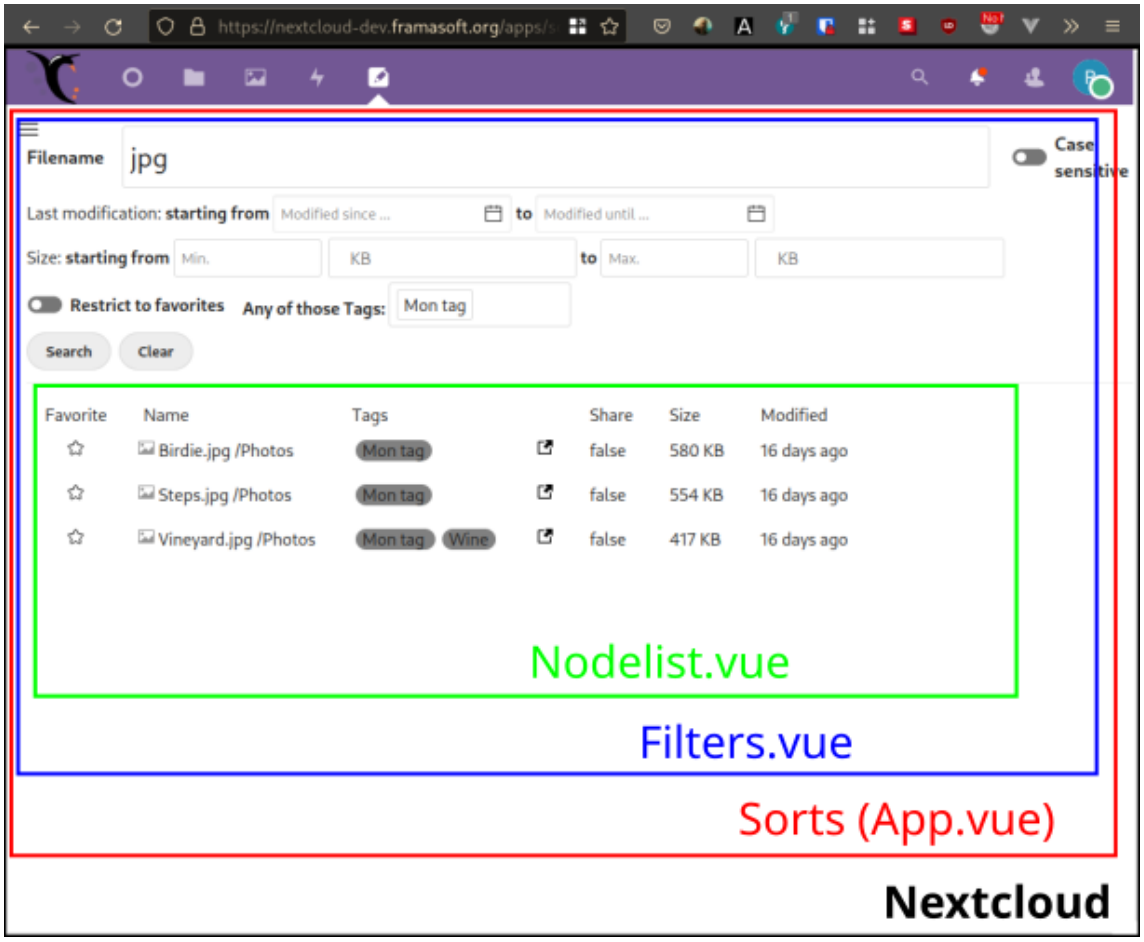


FIGURE 7 – Vue « filtres » de « Sorts »

Le détail de la division en composants du frontend sera abordé dans la section suivante (3.4.4)

3.4.4 Exemple de refactorisation

Un des intérêts majeurs de VueJS est la séparation du frontend en composants. Tout composant peut en instancier d'autres via une simple balise HTML portant le nom du composant à instancier.

Les composants parents passent des informations aux composants enfants via des directives `v-bind:variableEnfant="expressionParent"` : la `variableEnfant` est accessible au composant enfant et reflète à tout instant `expressionParent` qui est calculée dans le composant parent, et qui peut être une variable ou un appel à une méthode.

Les composants enfants peuvent à leur tour remonter de l'information à leur parent via des `events` : le parent enregistre à la création du composant une méthode à appeler lorsqu'un certain `event` est émis à l'aide d'une directive `v-on:monEvent="méthodeParent()"`. L'enfant peut ensuite émettre des événements via la fonction `emit()`, qui prend en argument le nom de l'évènement à transmettre et une variable optionnelle contenant des données à passer en paramètre à la `méthodeParent()`.

Vers mi-janvier, après un mois de développement, les composants du *plugin* étaient agencés comme sur la figure 8. Une première séparation en composants avait déjà était pensée lors du développement : le composant `App`, qui est le composant principal et le point d'entrée de l'application, charge les trois composants correspondants aux trois parties de l'interface utilisateur : le Breadcrumb, le formulaire de filtres et la liste des fichiers.

Le problème de cette structure est qu'elle concentre toute la logique, les données et la gestion d'états sur `Nodelist`. Il s'agit du composant qui récupère les fichiers de l'utilisateur dans un arbre (`nodesTree`) et qui les affiche, que ce soit des fichiers de l'explorateur de fichiers ou des fichiers résultant d'un filtre.

`Nodelist` a donc toutes les méthodes pour faire des requêtes au backend (pour demander le contenu d'un dossier ou pour demander les résultats d'une recherche), et le composant `Filters` n'est qu'un formulaire de choix des filtres, qui passe la sélection de l'utilisateur à `Nodelist` de manière indirecte (`Filters` émet un événement à `App`, qui *bind* la variable stockant cet événement au composant `Nodelist`).

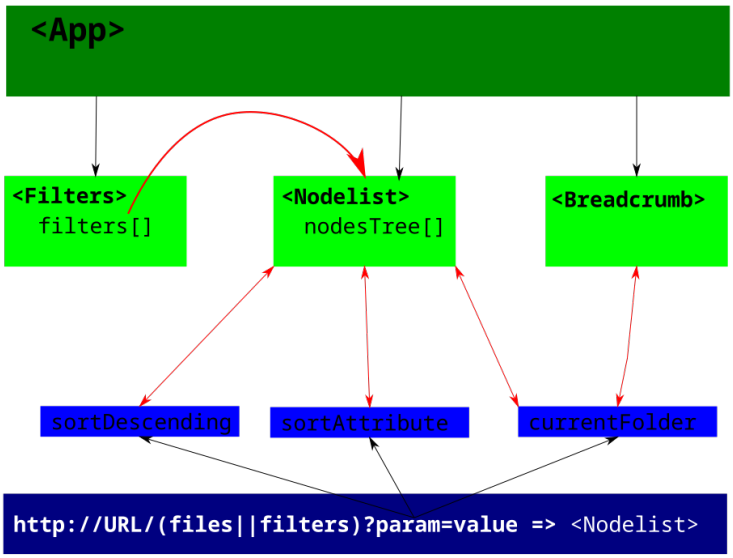


FIGURE 8 – Schéma des composasants de « Sorts » avant refactorisation

Cette structure est peu conventionnelle (deux composants découlant du même parent ne devraient pas avoir à s'échanger d'informations), et très peu maintenable. Une refactorisation était nécessaire pour repartir sur de bonnes bases et finir le développement plus aisément. J'ai donc modifié les composants pour atteindre l'organisation présentée à la figure 9.

`Nodelist` est ici dénué de toute logique, il ne s'agit plus que d'un élément de présentation auquel on `v-bind` un arbre de fichiers `nodesTree`, qu'il s'occupe d'aplanir et d'afficher (voir section 3.4.2), et qui émet des évènements lorsque l'on clique ou double-clique sur un dossier (déplier ou entrer dans le dossier). La récupération de l'arbre de fichiers `nodesTree` et sa modification est laissée aux composants instanciant `Nodelist`.

Le composant `Filters` contient désormais les méthodes pour requêter les fichiers filtrés en plus de présenter une interface utilisateur pour sélectionner ces filtres. Un nouveau composant `FileExplorer` a été introduit pour récupérer auprès du backend et modifier l'arbre des fichiers dans un contexte d'exploration des fichiers.

Le composant `App` détermine aussi l'état de l'application selon si l'URL commence par « files » ou « filters ». `FileExplorer` et `Breadcrumb` sont chargés par `App` si et seulement si l'URL commence par « files ». `App` passe aussi l'information à `Filters` via une directive *v-bind* afin qu'il affiche ou non sa liste de fichiers `Nodelist`. Ainsi, l'URL résulte en deux vues différentes : une vue « explorateur de fichiers », figure 6, et une vue « filtres », figure 7.

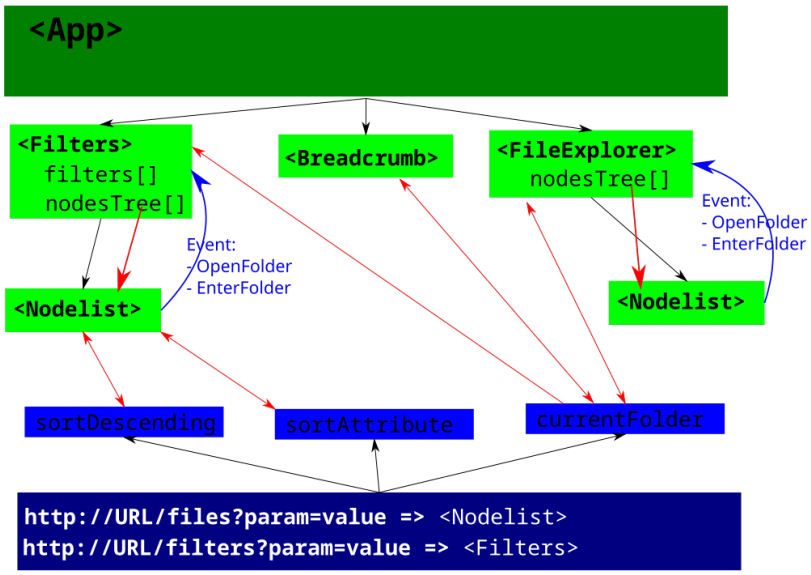


FIGURE 9 – Schéma des composants de « Sorts » après refactorisation

3.5 Autres tâches au sein de Framasoft

J'ai été chargé lors de mon stage d'autres tâches que le sujet principal évoqué ci-dessus.

J'ai travaillé autour du logiciel « Yakforms », un logiciel de réalisation de formulaires créé par Framasoft et dont nous souhaitons que le développement soit repris par la communauté. Beaucoup de travail a été fait dans ce sens par Théophile Lemarié, ancien élève du département Télécommunication de l'INSA de Lyon, lors de son stage de fin d'études en 2020. J'ai donc eu pour tâche ponctuelle d'éprouver la documentation rédigée par Théophile en installant le logiciel et en remontant les erreurs et difficultés non couvertes par la documentation du processus d'installation.

J'ai aussi participé de manière régulière d'octobre à novembre au support technique du service de Framasoft reposant sur ce logiciel. Cette expérience m'a permis de voir d'autre facette du travail des salariés de Framasoft et de me familiariser avec les infrastructures hébergeant les services de Framasoft et leur fonctionnement.

Il était aussi prévu que je tente de développer un prototype d'intégration au frontend Nextcloud d'une librairie de « tutoriel » JavaScript. L'idée derrière cette tâche était de juger de la faisabilité technique de créer une « visite guidée » de l'interface web de Nextcloud, qui était un des besoins repérés lors de l'enquête réalisée auprès de structures militantes.

Ce sujet était prévu comme développement annexe au cas où le développement de « Sorts » soit plus rapide que ce que nous avions estimé. Comme le temps de développement de Sorts fut conséquent, je n'ai finalement pas abordé cette tâche lors de mon stage.

4 La maîtrise du numérique et l'impact des organisations alternatives

L'enjeu du projet « Framacloud », au sein duquel se déroule mon stage, est d'outiller numériquement les acteurs du progrès social et de la justice sociale [1] afin de participer à leur capacitation et d'accroître leurs impacts sur la société.

Cet objectif découle de plusieurs intuitions. Premièrement, que la maîtrise des outils numériques est un facteur important pour l'impact d'un groupe sur la société et deuxièmement, que les structures du progrès social et de la justice sociale sont défavorisées dans leurs usages et accès au numérique. De plus, notre choix de répondre à cette problématique avec des actions autour de la collaboration numérique est-elle justifiée ? Ces intuitions sont-elles fondées et quelle place a la collaboration numérique dans la capacitation numérique des acteurs visés ?

4.1 Quel impact sur la société peut-on avoir en maîtrisant le numérique ?

Dans l'introduction de Culture Numérique, Dominique CARDON compare la rupture profonde entraînée par l'avènement du numérique avec l'invention de l'imprimerie au XV^{e} siècle et la révolution culturelle qui s'ensuivit [7]. Il va sans dire aujourd'hui que le numérique a pris une place centrale dans la société et la vie quotidienne : télétravail, divertissements numériques, démarches en ligne, achats en ligne, ...

Quels impacts politiques peut donc avoir la maîtrise du numérique sur une société de plus en plus numérisée ?

Un premier angle d'observation est celui de la libération de l'expression populaire via internet qui, selon Dominique CARDON [8] et Dominique BOULLIER [9], a permis l'apparition d'une nouvelle forme de démocratie.

Cette nouvelle « démocratie internet » résulte notamment du poids médiatique des réseaux sociaux, qui permettent à tous de produire et de commenter, sur un supposé pied d'égalité, opinions, actualités ou revendications. Ces réseaux, bien qu'investis par les journaux traditionnels et les personnalités politiques, confèrent un poids parfois supérieur aux individus sans « capital de notoriété public » préalable (que ce capital soit militant, politique ou médiatique).

Cette nouvelle forme de démocratie se concrétise via des mouvements sociaux populaires d'envergure comme celui du Printemps Arabe ou celui des Gilets Jaunes. Ce dernier s'est construit via les réseaux sociaux et notamment la plateforme Twitter

1. Par ce terme d'acteurs, nous souhaitons englober un ensemble vaste d'organismes : des associations, des syndicats, des collectifs informels, des entreprises, ...

et est assez caractéristique de ces nouveaux mouvements : refus de hiérarchisation (et notamment le refus de désigner des portes-paroles) et grande diversité d'opinions au sein de ses membres [10].

Ces nouveaux mouvements, pour lesquels la maîtrise des réseaux sociaux a été un élément clef, ont eu des impacts politiques importants et sont un exemple de l'impact politique de la maîtrise du numérique.

Cependant, un deuxième constat vient mitiger ces effets positifs du numérique pour le changement social. Selon le livre « The revolution that wasn't » de la chercheuse en sciences politiques Jen SCHRADI, cité par Le Monde dans l'article « Internet est-il de droite » [11], les groupes militants de droite et d'extrême droite ont une plus grande présence sur internet et sont aussi les plus hiérarchisés, ce qui leur permet d'avoir une plus grande visibilité et de diffuser plus rapidement et fortement leurs idées.

L'usage du numérique est donc un levier important pour le militantisme et peut mener à des mouvements sociaux de grande ampleur, mais internet est une scène dominée par des thématiques conservatrices. Est-ce que les structures du progrès social et de la justice sociale sont défavorisées dans leur accès au numérique ?

4.2 Les structures alternatives ont-elle accès à la collaboration numérique ?

Sur quels aspects les structures de la justice sociale et du progrès social pourraient-elles avoir des difficultés d'accès au numérique ?

Guillaume GARCZYNSKI, dans son article « Fracture numérique, Fracture sociale », met en avant le fait que de nombreuses catégories de personnes défavorisées sont tenues à l'écart du numérique. Ces freins à l'inclusion peuvent être financiers (personnes touchant des minimas sociaux, sans-papiers, ...), ou liés à un illettrisme numérique (personnes âgées, ruraux, jeunes n'ayant qu'un usage récréatif du numérique, ...). Cette fracture numérique est la conséquence d'une fracture sociale et cause une fracture sociale. Elle touche principalement des publics visés par les structures du changement social ou susceptibles de militer dans ces structures.[12]

Les mouvements sociaux évoqués dans la section ci-dessus nuisent aussi la dimension collective des structures classiques des luttes sociales (syndicats, associations, partis, ...) au profit d'une individualisation des prises de positions. Cette individualisation peut nuire aux luttes sociales en fragmentant les voix et les revendications [13] [10]

Est-ce qu'il ne manquerait pas une maîtrise de la collaboration numérique pour permettre aux luttes sociales de recréer du collectif et de la convergence dans un contexte de numérisation de la société ?

C'est en tout cas sur ce créneau que souhaite se poser le projet Framacloud en utilisant le logiciel Nextcloud comme prétexte pour créer de la collaboration entre les structures du progrès social et de la justice sociale.

Outre les difficultés numériques individuelles, le problème du numérique collaboratif est qu'il nécessite une infrastructure et une organisation plus importante que le simple accès aux réseaux sociaux. Il faut choisir les services via lesquels collaborer, que les militants apprennent à les utiliser, que leur usage devienne habituel et que des processus de collaboration soient réfléchis et mis en place selon les besoins de chaque structure. Ces prérequis expliquent que parmi les associations françaises, 39% seulement des associations utilisent le numérique pour mieux collaborer, majoritairement des associations qui se déclarent à l'aise avec le numérique [3].

De plus, les outils de collaboration les plus faciles d'accès aujourd'hui sont les grandes solutions centralisées des entreprises du capitalisme de surveillance, comme la « suite Google ». Ces outils soulèvent de nombreuses inquiétudes quant à leur utilisation par un public militant qui décrit comme frein principal à un usage associatif du numérique collaboratif l'inquiétude que les données personnelles ne soient pas suffisamment protégées [12].

Il y a donc une urgence, si l'on prend le point de vue des forces progressistes plutôt que conservatrices, à accompagner les structures militantes vers une plus grande maîtrise des processus de collaboration numérique.

4.3 Les limites de la numérisation des luttes

Dans la course au militantisme numérique, il y a une urgence à outiller les structures militantes du changement social. Mais quelle est l'importance de cette scène numérique par rapport au monde non-numérique ?

Les militants pour le progrès social et la justice sociale seraient plus focalisés sur des actions de terrains concrètes que sur de la communication, et notamment de la communication numérique [11].

Par exemple, l'accompagnement et l'aide aux populations défavorisées ne peuvent pas être remplacés par des interactions numériques. Le numérique ne peut, dans ce cadre, suppléer à l'action physique et est limité à la sensibilisation du public à ces problématiques ou à la coordination des militants [13].

Le militantisme de terrain a aussi un rôle de catalyseur dans l'engagement. Il est souvent à l'origine de l'engagement sur le long-terme de nombreux individus dans le monde du militantisme [13]. Mais aussi, les militants actifs en ligne seraient les militants déjà sensibilisés et investis en présentiel dans les structures militantes [7].

Le numérique a modifié le militantisme pour lequel il est une nouvelle porte

d'entrée ou un nouvel outil, mais il ne remplace pas l'action militante physique.

À l'aune de ces éléments, il semble possible d'établir une corrélation forte entre maîtrise des usages numériques (notamment collaboratifs) d'une part, et capacité d'impact social et politique des structures d'autre part. Cependant, il apparaît aussi que la maîtrise de ces outils et techniques serait une condition nécéssaire, mais non suffisante pour que les structures œuvrant pour le progrès et la justice sociale puissent voir leurs idées et leurs actions l'emporter sur celle des mouvements plus conservateurs.

5 Métier : Développement web *Full-Stack* sur une application Open-Source

Framasoft étant une petite structure associative de dix salariés, formant une seule équipe où chacun travaille sur des projets différents, les postes et les processus de gestion d'équipe s'éloignent un peu de ce qui se fait dans de plus grosses structures entrepreneuriales.

Par exemple, le métier de mon tuteur de stage est celui de codirecteur de Framasoft. C'est un poste très transversal, mêlant coordination d'équipe, définitions des stratégies de développement et de communication de Framasoft et représentation de l'association auprès des médias et partenaires.

Néanmoins, on peut tirer quelques ressemblances entre le rôle que j'ai mené pendant ce stage et des postes plus « classiques » du monde du travail.

La partie technique de mon stage pouvait se rapprocher du travail de développeur web *Full-Stack*, avec en plus un aspect « gestion de projet » qui m'était laissé libre. Ceci rapproche un peu mon poste des postes de gestion de projets techniques, modulo les parties liées à la gestion d'équipe.

Néanmoins, le domaine dans lequel il y a le plus de ressemblances à tirer est celui de l'Open-Source. Développer un logiciel avec des collaborateurs d'horizons divers, amateurs où professionnels, d'entreprises et de nationalités différentes, travaillant quasi-exclusivement à distance, est une des caractéristiques forte de ce mouvement.

J'ai pu, au cours de ce stage, évoluer dans un tel contexte en participant au développement de Nextcloud. J'ai aussi créé et géré un projet Open-Source avec le *plugin* « Sorts », ce qui m'a amené à échanger avec des développeurs de l'entreprise allemande Nextcloud GmbH à plusieurs occasions, mais aussi à apprendre comment gérer les contributions logicielles.

En plus de mes convictions libristes, le contexte de travail du développement Open-Source m'ont beaucoup plu et je souhaite poursuivre ma carrière dans ce milieu. J'envisage actuellement de postuler sur des postes de développement backend, et notamment au sein de Nextcloud GmbH. Ce type de poste concilierait bien mon attrait pour le développement backend, pour les problématiques techniques, et pour les *process* liés au développement Open-Source.

6 Retour d'expérience : un stage à Framasoft

Techniquement j'ai approfondi les compétences de développement web que j'avais acquises lors de ma formation avec la prise en main de PHP et VueJS qui comptent parmi les technologies majeures du développement web. J'ai aussi énormément progressé sur ma lecture et ma compréhension du code source d'un « vrai » logiciel professionnel. J'entends par là que l'ampleur du code source de Nextcloud (environ 6 millions de lignes de code) est bien plus grande que tout ce que nous abordons lors de note formation et la compréhension du fonctionnement d'un logiciel s'étalant sur plusieurs centaines de fichiers demande de comprendre les mécanismes « d'articulation » des différentes parties du code source.

Le développement de « Sorts » a aussi été l'occasion de faire de la gestion de projet. J'ai notamment repris plusieurs fois mon planning, la décomposition de mon développement en fonctionnalités et j'ai dû passer par plusieurs étapes de refactorisation. Ce tâtonnement est à la fois dû au fait que j'apprenais les technologies que j'utilisais au fur et à mesure mais aussi parce que j'ai à plusieurs reprises fait des erreurs d'estimation que je saurais aujourd'hui éviter.

Ce stage m'a aussi permis de revoir mon approche au télétravail et j'ai observé une progression sur ma capacité à compartimenter mon attention et à rester concentré sur mon travail depuis chez moi. Premièrement, je travaillais sur des projets Open-Source avec des développeurs externes à Framasoft ou des collègues en télétravail (plus de la moitié de l'équipe salariée de Framasoft n'est pas sur Lyon). Deuxièmement, je travaillais le lundi et le vendredi de chez moi et le reste du temps depuis les bureaux de Framasoft aux Locaux Motiv', qui est un espace de travail partagé pour les structures de l'économie sociale et solidaire.

Ce fonctionnement hybride s'est avéré être un vrai avantage pour ma motivation et mon efficacité. Ma présence aux Locaux Motiv' m'a permis de profiter d'une ambiance de travail riche en interactions humaines ainsi que d'un cadre bien séparé entre travail et vie privée, et mes journées de travail à distance m'ont autorisé une plus grande flexibilité sur mes horaires et de gagner sur mes temps de trajet domicile/travail. J'aimerais dans mes futurs emplois pouvoir retrouver ce mode de fonctionnement.

Enfin, ce stage était aussi pour moi une première expérience de collaboration avec des acteurs variés dans un contexte international, que ce soit via des interactions sur GitHub lorsque j'ai contribué au développement de Nextcloud ou lors de la réunion en visioconférence, en anglais, avec des ingénieurs allemands de Nextcloud GmbH.

Pour conclure, je me sens aujourd'hui en mesure d'entrer dans le monde du travail salarié, de m'investir dans des projets longs et d'aborder le développement logiciel au niveau professionnel.

7 Conclusion

Mon stage ayant pris place sur le début du projet « Framacloud », il reste beaucoup de travail, d'exploration et de projets à mettre en œuvre pour atteindre l'objectif principal : faire de Nextcloud un levier de la collaboration entre les structures militant pour le progrès et la justice sociale.

Pour ce qui est du *plugin* « Sorts », l'objectif des prochaines semaines sera d'en publier une première version. Il s'agira de corriger ses dernières erreurs, de le rendre le plus accessible possible en « lissant » l'interface utilisateur un peu austère et de le rendre public via un communiqué sur le site de Framasoft et une publication du *plugin* sur le site officiel des *plugins* Nextcloud.

Par la suite, l'équipe salariée de Framasoft étant volontairement limitée à 10 personnes et travaillant déjà sur plusieurs dizaines de projets en parallèle, il n'est pas prévu de rajouter du temps de développement salarié sur « Sorts ». Un suivi du *plugin* sera néanmoins réalisé, avec deux objectifs : garder le *plugin* actuel fonctionnel et récolter des retours d'expériences afin d'affiner notre compréhension des besoins des utilisateurs ciblés par le projet « Framacloud ». Les développements futurs seront donc laissés à la charge de la communauté ou à d'éventuels futurs sujets de stage. Je souhaite également rester membre bénévole de l'association et *maintainer* du *plugin* sur mon temps libre. Je serai donc en mesure de suivre les correctifs et potentiels développements.

En ce qui concerne le reste du projet Framacloud, « La route est longue, mais la voie est libre » [1]. Les chemins pour atteindre l'objectif du projet sont multiples : il s'agira de continuer à améliorer la solution technique par des développements logiciels, de communiquer sur le projet, d'accompagner les structures ciblées et de proposer une offre Nextcloud à toute association militante qui en exprimerait le besoin [4].

Afin de mener à bien ces missions, il va nous falloir comprendre les besoins et attentes de nos utilisateurs cibles. C'est un travail qui a été commencé avec mon stage et le futur du projet pourra se baser sur mon rapport d'étonnement et les résultats de l'enquête, mais aussi de nouvelles ressources comme la récolte de retours utilisateurs sur le déploiement du *plugin* « Sorts » et l'augmentation de l'offre Nextcloud. Il s'agira aussi de collaborer avec Nextcloud GmbH afin de comprendre leur vision du futur du logiciel et les points sur lesquels on peut améliorer son usage associatif.

Le projet Framacloud, avec l'ampleur actuellement prévue, devrait durer 3 ans [4] et arriver à terme en 2024/2025. Le projet est encore jeune et ses objectifs peuvent donc encore s'affiner au gré des avancements et des problématiques découvertes dans les mois à venir.

1. Devise non-officielle de Framasoft

Références

[1] FRAMASOFT. *Statuts de l'association Framasoft*. 2021. URL : https://soutenir.framasoft.org/sites/default/files/statuts-Framasoft-2021.pdf.

[2] FRAMASOFT. *Rapport d'activité 2020*. 2021. URL : https://soutenir.framasoft.org/sites/default/files/Framasoft_rapport_activite_2020-1.0.pdf.

[3] SOLIDATECH et RECHERCHESOLIDARITÉS. *La place du numérique dans le projet associatif en 2019*. 2019. URL : https://recherches-solidarites.org/wp-content/uploads/2019/12/Num%C3%A9rique-Rapport-2019.pdf.

[4] FRAMASOFT. *Préprojet Framacloud : Software to the people !* 2021.

[5] VARIOUS. *Nextcloud developer documentation*. version 23. Nextcloud GmbH. 2022. URL : https://docs.nextcloud.com/server/23/developer_manual.

[6] VARIOUS. *Nextcloud PHP API*. latest version. Nextcloud GmbH. 2022. URL : https://nextcloud-server.netlify.app/namespaces/ocp.html.

[7] Dominique CARDON. *Culture Numérique*. Paris : presses de Sciences Po, 2016.

[8] Dominique CARDON. "L'espace publique numérique". In : *Culture Numérique*. Paris : presses de Sciences Po, 2016.

[9] Dominique BOULLIER. "Sociologie politique du numérique". In : *Sociologie du numérique*. Armand Colin, 2016. ISBN : 978-2-200-29165-5.

[10] Natacha SOUILLARD et al. "Les Gilets jaunes, étude d'un mouvement social au prisme de ses arènes médiatiques". In : *Terminal, Technologie de l'information, culture et société* 127 (2020). DOI : 10.4000/terminal.5631.

[11] INTERNETACTU. *Internet est-il de droite ?* 2019. URL : https://www.lemonde.fr/blog/internetactu/2019/05/08/internet-est-il-de-droite/.

[12] Guillaume GARCZYNSKI. "Fracture numérique, fracture sociale". In : *Revue Projet* N° 371, Internet réinvent-t-il le militantisme (2019). DOI : 10.3917/pro.371.0033.

[13] Anaïs THEVIOT. "Le militantisme, cinquante ans après Mai 68". In : *Revue Projet* N° 371, Internet réinvent-t-il le militantisme (2019). DOI : 10.3917/pro.371.0037.

Glossaire

fork Base de code source créé par un développeur à partir du code source officiel (upstream) dans le but d'y développer une *feature* (à proposer à l'upstream pour intégration) OU Nouveau logiciel créé sur les bases du code source d'un logiciel préexistant.. 8, 9, 13, 15

merge request Demande d'intégration d'un nouveau morceau de code au code source d'un logiciel. 12, 13, 19

Mobilizon Alternative libre aux évènements Facebook. 5, 6

Nextcloud GmbH Entreprise allemande à responsabilité limité (GmbH) motrice du développement du logiciel Nextcloud. 8, 11, 15, 16, 18, 25, 36, 37

PeerTube Logiciel libre de distribution de vidéos, client-serveur et pair-à-pair. 5, 6

upstream Le code source officiel du logiciel OU un logiciel dont découle un second logiciel (fork). 13, 15

Acronymes

CHATONS Collectif des Hébergeurs Alternatifs, Transparents, Ouverts, Neutres et Solidaires. 5

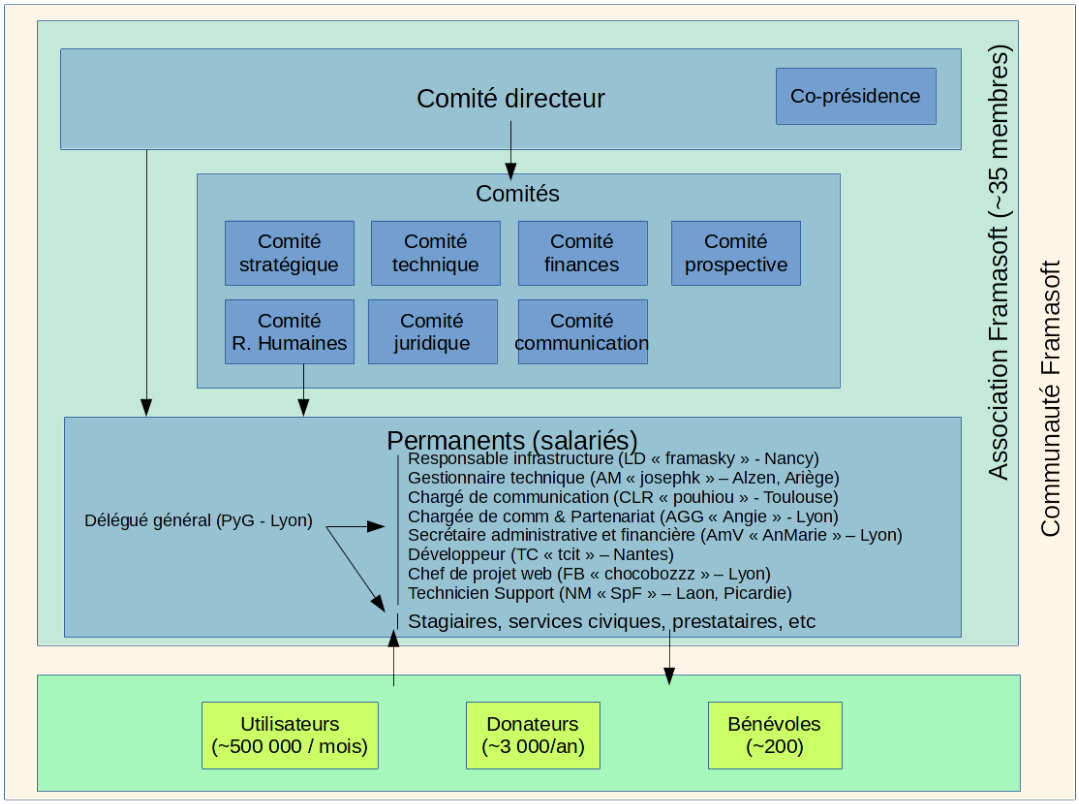
GAFAM Google, Amazon, Facebook, Apple et Microsoft. 4

LAMP Linux, Apache, MySql, PHP (pile logicielle classique d'un serveur web). 9

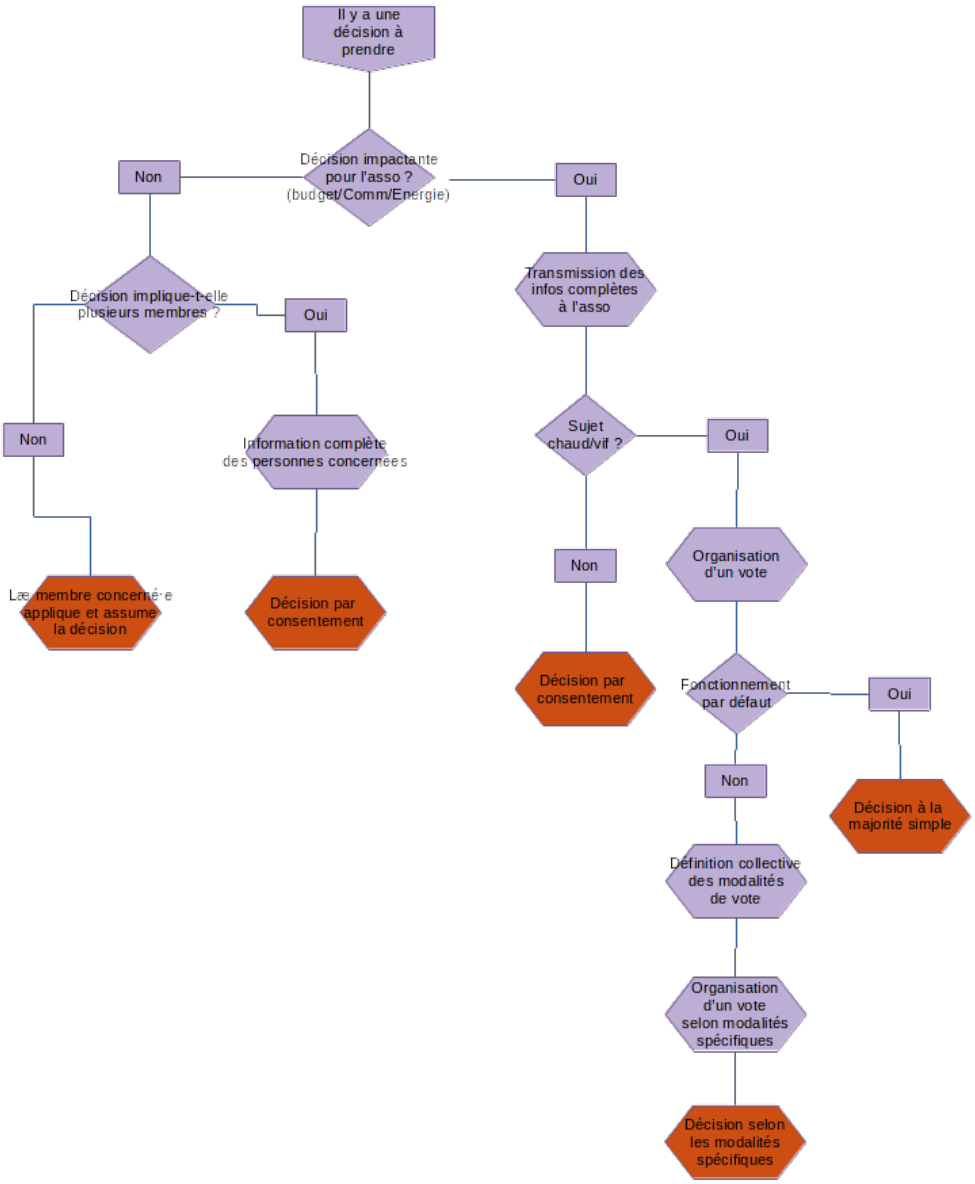
SWOT Strengths, Weaknesses, Opportunities, Threats (Forces, Faiblesses, Opportunités, Menaces). 3, 7

Annexes

A Organigramme de l'association



B Modalités de prises de décisions à Framasoft



C Gantt de déroulement du stage