Dynamique de développement des communautés du logiciel libre

Conditions d'émergence et régulations des tensions

Didier Demazière, François Horn, Marc Zune*

La renommée et l'usage des logiciels libres ne cessent de croître et le nombre de projets déposés sur les sites de développement mutualisés poursuit une croissance exponentielle. Le site de développement communautaire SourceForge dénombrait, fin 2005 plus de 100 000 projets et plus d'un million de développeurs référencés. Ces projets sont marqués par une très forte hétérogénéité concernant le nombre de contributeurs et leur organisation. Si les développeurs de logiciels libres et certains projets emblématiques ont fait l'objet d'études, une question reste cependant, à ce jour, peu analysée : qu'est-ce qui explique que parmi tous ces projets certains parviennent à attirer des contributeurs permettant d'enclencher une dynamique de développement ?

Parmi l'ensemble des modèles de production de logiciels [Horn, 2004], le développement des logiciels libres présente l'originalité d'être fondé sur un mode d'organisation coopératif qui s'appuie largement sur les commodités organisationnelles issues d'Internet. Les développeurs ne sont pas inscrits dans la même organisation, sont dispersés, ont des relations médiatisées par le réseau Internet, ne sont pas reliés par les fils d'un organigramme quelconque. Il n'existe pas d'interactions directes, codifiées et prescrites entre les producteurs du logiciel. En principe, toute personne peut fournir une contribution et participer à l'élaboration d'un bien collectif et public.

Les échanges entre contributeurs s'effectuent quasi-exclusivement à partir de modes de communication électroniques : courrier électronique, listes de diffusion, forums, *chat*, dispositif d'enregistrement et de visualisation du code produit, etc. Ce fonctionnement à distance, permet d'attirer des contributions nombreuses, favorise un meilleur repérage des erreurs par les utilisateurs, et autorise des corrections plus rapides. Cette configuration conduit à s'interroger sur les caractéristiques de l'action collective qui permet de passer d'engagements individuels volontaires, et potentiellement volatiles et instables, à la réalisation d'une production collective, impliquant continuité et pérennité. Le mode de développement du logiciel libre qui

* Didier Demazière (Printemps, CRS/Université de Versailles - Saint-Quentin-en-Yvelines) ; François Horn (CLERSE-IFRESI, Université Lille 3 ; Marc Zune (CSTEF, FNRS/Université Libre de Brxelles).

repose largement, en apparence, sur le caractère bénévole des contributions, se singularise par rapport aux modes de production habituels encadrés par des relations de travail salariales ou contractuelles. Mis à part une licence d'utilisation spécifique, cette activité n'est en effet régulée par aucun dispositif juridique relevant du droit du travail ou commercial, ni par des incitants pécuniaires, ni par des rapports de subordination [Gensollen, 2004]. Les groupes de production sont d'ailleurs désignés dans les milieux du logiciel libre par le terme communauté, et la gestion est référée à un "modèle du bazar" [Raymond, 1998], fondé sur une très forte décentralisation auprès de contributeurs non sélectionnés, dépourvu de coordination formelle, de cahier des charges ou d'injonctions temporelles, laissant libre cours à l'imagination et l'implication individuelles [Glass, 2003, Holgrewe et Werle, 2001].

Cette modélisation du fonctionnement des communautés de logiciels libres a une fonction politique consistant à affirmer le caractère original et alternatif de ce mode de production par rapport à l'industrie du logiciel propriétaire ("modèle de la cathédrale"). Sa valeur descriptive et sociologique apparaît en revanche contestable. Car la production d'un bien collectif de qualité à partir de contributions individuelles volontaires nécessite des formes de régulation. Développer un logiciel selon un mode collaboratif suppose de réaliser une série d'opérations : mobiliser des développeurs autour d'un projet, assurer une relative continuité des participations, vérifier la qualité des contributions, agencer les développements validés en un produit cohérent et opératoire, etc.

Notre analyse vise à tenter de cerner le fonctionnement des communautés de logiciels libres dans leurs processus de constitution et de développement. Au-delà de l'extrême diversité organisationnelle des différents projets de développement de logiciels libres [Demazzière, Horn, Jullien, 2003], il s'agit d'éclaircir plusieurs interrogations : comment se constituent et se structurent ces collectifs de travail ? Comment sont-ils animés, gérés, fédérés, soutenus ? Comment maintient-on, dans ces conditions, un produit cohérent, avec une certaine identité et continuité dans le temps ? Pour cela, nous proposons de spécifier une série de moments-clés dans le développement d'un projet libre, du stade de l'émergence aux possibilités d'extinction ou de scissions. À cet effet, nous nous appuyons sur deux études de cas de logiciels libres, les projets Spip et Claroline[1].

Conditions d'émergence d'une communauté

La brève présentation en encadré des deux projets à la base de notre réflexion nous renseigne d'emblée sur une première condition nécessaire au

Suite page 74

1. Ceux-ci ont été choisis parce qu'ils sont de taille moyenne, sont en phase de développement et sont confrontés à des enjeux de légitimité que les projets les plus connus et les plus établis (Linux, Apache, etc.) ne connaissent plus. L'investigation empirique a été menée à la manière d'une enquête ethnographique en s'appuyant sur 35 entretiens approfondis de recherche auprès de participants présentant des contributions et activités différenciées.

SPIP, UN OUTIL D'AUTOPUBLICATION SUR LE WEB

Le projet "Spip" se présente comme une figure *a priori* emblématique d'un fonctionnement communautaire fondé sur l'apport central d'individus dispersés et non reliés par quelque arrangement institutionnel ou d'emploi particulier. C'est afin d'aider les membres d'une association à s'exprimer par eux-mêmes en toute autonomie que l'un d'entre eux a l'idée de développer un outil informatique facilitant la fabrication de pages Web et la gestion du site de l'association.

La particularité est que l'usage de ce logiciel ne nécessite pas de connaissances techniques particulières et doit pouvoir "tourner" sur des serveurs peu véloces, et par conséquent peu onéreux. Après quelques mois d'usage interne, la demande se manifeste rapidement pour que cet outil puisse être utilisé par d'autres associations. Un sympathisant informaticien se propose d'aider le concepteur de l'outil, infographiste, et s'implique dans le développement. En même temps, un webmestre d'un grand journal mensuel d'opinion manifeste l'intérêt d'utiliser ce logiciel à des fins de publication en ligne. Cette équipe de base constituera le "noyau" historique du projet, les "auteurs" du logiciel.

Ces trois fondateurs partagent une ligne philosophique qu'ils souhaitent poursuivre, celle de l'auto-publication et de l'indépendance des utilisateurs face à toute contrainte à la liberté d'expression. Ils entendent, face au péril pressenti d'une certaine mainmise des industries du contenu, préserver des espaces de libre expression pouvant être investis par tout acteur de la vie sociale, quelles que soient ses compétences techniques et ses ressources matérielles. Cette orientation, traduite en manifeste du Web indépendant, se décline à la fois dans des partis pris techniques et fonctionnels (liberté de réaction aux contenus publiés, limitation de la hiérarchie des droits de publication, etc.), mais également dans un projet militant de diffusion de la liberté d'expression.

Le succès du logiciel est fulgurant. Spip connaît une utilisation fortement croissante, au point de devenir une référence incontournable parmi l'offre de logiciels de gestion de contenu sur Internet. C'est que l'outil développé, certes imparfait dans ses premières moutures, répond à un besoin criant au moment de son apparition, face aux solutions "propriétaires" souvent trop onéreuses et anglophones. Ce succès s'accompagne du développement d'une communauté importante et internationale du fait de la traduction du logiciel et de son support en plus de trente langues. Ces contributeurs présentent des profils diversifiés : militants de sites associatifs, webmestres ,infographistes, informaticiens pointus, que ce soit à titre professionnel ou d'activité hors travail.

Après une année de développement, un prestataire informatique sous contrat avec une administration publique propose une réécriture partielle du logiciel en introduisant une méthode de programmation plus complexe et l'ajout de fonctionnalités jugées plus adaptées au monde professionnel. L'absence de collaboration avec la communauté provoquera le rejet de la greffe proposée et une scission de fait (un *fork* dans la terminologie indigène), avec le développement en parallèle d'une autre version du logiciel intitulée Spip-Agora.

CLAROLINE, UNE PLATE-FORME DE *E-LEARNING*

Claroline est un logiciel développé par une université belge. L'idée de ce projet revient à un jeune docteur en philosophie chargé par son institution de promouvoir l'utilisation d'un logiciel commercial de support à l'enseignement en ligne. Malgré les efforts, notamment de formation, la mission est un échec, le logiciel étant jugé trop lourd et contraignant par les enseignants. Dans le même temps, l'initiateur du projet découvre sur Internet des outils logiciels libres simples répondant aux principales fonctionnalités demandées par les enseignants. Il décide alors d'intégrer ces outils et de les doter d'une interface commune créant ainsi un nouveau logiciel qu'il diffuse sous licence libre. Claroline connaît un succès rapide au sein de l'institution et dans d'autres universités aux besoins identiques. En interne, l'équipe de développement s'étoffe progressivement, avec deux autres développeurs engagés à temps plein pour répondre aux besoins des utilisateurs locaux et améliorer les fonctionnalités demandées. Des utilisateurs avancés, surtout des gestionnaires de campus numériques d'autres universités (plus d'une centaine, essentiellement du sud de l'Europe et du monde hispanique), participent aux discussions sur les forums du projet, proposent des traductions, rapportent des bogues lors de chaque nouvelle version, font part de leurs souhaits en matière de fonctionnalités nouvelles.

L'équipe interne défend deux valeurs centrales à la base du projet : la flexibilité et la simplicité d'utilisation, à l'exact opposé des traits attribués aux logiciels commerciaux les plus en vogue. Il s'agit de proposer à l'enseignant un ensemble d'outils de support à l'enseignement (forum, dépôt de documents, agenda, etc.) sans le contraindre à un usage prescrit. Par conséquent, toute contribution au code du logiciel, indépendamment de sa qualité technique, n'est acceptée que pour autant qu'elle respecte ces deux traits. Dans les faits, bon nombre de propositions de collaboration sur le code sont ainsi rejetées. Les apports techniques proposés par les contributeurs "distants" portent davantage sur des petites corrections, qui, lorsqu'elles sont jugées intéressantes, sont réécrites au moins en partie par les développeurs de l'université. L'essentiel des autres contributions portent par conséquent sur des activités de traduction ou de rapports de bogues.

La grande diffusion du logiciel entraîne, assez rapidement, une multitude de demandes de prestations de services visant à adapter le logiciel aux contextes spécifiques des utilisateurs. Cette demande amène les fondateurs à solliciter une aide publique afin d'améliorer les fonctionnalités du logiciel. Cet argent public permettra, en 2004, d'étendre l'équipe interne à trois nouvelles personnes. Cependant un conflit éclate entre le fondateur du logiciel et l'université qui veut lui retirer le *leadership* du projet. L'initiateur du projet décide alors de créer sa propre société de conseil en organisant son propre *fork* de Claroline, intitulé Dokeos.

Suite de la page 72

développement d'une communauté : l'outil proposé comble un manque dans l'offre des logiciels commerciaux ou répond à des besoins mal couverts par ces derniers étant donné l'impossibilité de les améliorer vu l'absence d'accès

au code source. Ainsi les logiciels libres les plus connus et les plus utilisés sont nés sur des créneaux où leurs équivalents commerciaux n'existaient pas (les outils logiciels nécessaires au fonctionnement d'Internet dans les années 80) ou dans des domaines où les qualités des logiciels existants étaient jugées insuffisantes, notamment en termes de sécurité et de fiabilité (par exemple les systèmes d'exploitation). Des manques plus limités des logiciels courants peuvent cependant suffire pour créer un espace propice à la naissance d'un logiciel libre, comme l'absence d'une version francophone, la nécessité de ressources importantes et donc coûteuses ou la simplicité d'utilisation.

En effet, les logiciels commerciaux sont souvent des produits très complexes marqués par une culture informaticienne. Il peut en résulter un décalage par rapport aux besoins d'une partie des utilisateurs, ce qui explique que l'on trouve fréquemment dans les noyaux initiateurs des logiciels libres des non-informaticiens professionnels. Les cas de Spip et de Claroline sont à cet égard emblématiques de la diversité des origines disciplinaires des initiateurs des projets : un pilote de ligne reconverti dans l'infographie, un docteur en mathématiques et un ingénieur télécom dans le cas de Spip, deux philosophes et un pédagogue dans le cas de Claroline.

Outre le fait de combler des manques existants, les choix effectués par les initiateurs du projet définissent l'identité originale [Salais, Storper, 1993] du logiciel qui sera développé. Les caractéristiques particulières du logiciel sont la traduction de cette identité spécifique. Ces caractéristiques qui sont dépendantes du processus de production du logiciel et qui influent sur son utilisation peuvent apparaître de prime abord comme de simples choix techniques entre des exigences contradictoires (richesse fonctionnelle *versus* simplicité d'utilisation…). Mais il est fréquent qu'en réalité les options retenues soient marquées par des conceptions sociales. C'est particulièrement net quand le logiciel vise à informatiser une activité existante : dans ce cas, la définition des caractéristiques fonctionnelles du logiciel est le reflet direct de la vision de l'activité. Ainsi, les caractéristiques de Spip sont fortement dépendantes de la représentation des initiateurs de ce qu'est une activité éditoriale et de la manière de la mener. L'interface graphique de la zone d'administration du logiciel mène à une mise en scène du processus de rédaction, qui passe notamment par la discussion et la validation de pairs avant publication. Placé dans l'espace public, l'article rédigé est associé à un dispositif de forum afin de susciter commentaires, réactions voire controverses.

Cette stimulation du regard des pairs vise à insuffler et promouvoir une culture de débats dans la publication sur Internet. De même, Claroline se définit par le caractère simple et incrémental de son utilisation, qui renvoie à un modèle de pédagogie active. Ce modèle tend à considérer la technologie comme miroir des processus d'apprentissage humains "naturels" qui dépassent la simple acquisition de connaissances ou de règles. Ceux-ci s'inscrivent également dans des contextes, attitudes et comportements culturellement situés [Lebrun, 2002]. Le logiciel doit servir les choix du pédagogue, faciliter les processus d'apprentissage à partir d'expériences diverses plutôt que

d'enfermer un transfert de connaissance dans un *one best way* pédagogique universel. L'ensemble des choix effectués à partir des orientations symboliques et idéologiques des promoteurs du projet va définir l'identité spécifique du logiciel proposé et sa différenciation par rapport à d'autres produits voisins et concurrents. Quand cette identité répond aux attentes d'utilisateurs, et même si le logiciel est très imparfait dans ses premières moutures, une communauté de contributeurs peut émerger. En effet, cette identité spécifique, discriminante par rapport aux autres logiciels existants est le vecteur d'un sentiment d'appartenance communautaire qui peut motiver une partie des utilisateurs à devenir des contributeurs.

En outre, c'est sans doute parce qu'on ne peut réduire l'activité de développement à la réalisation d'un simple outil, mais que celui-ci s'inscrit dans une perspective temporelle indéterminée d'un projet, qu'une communauté peut se constituer et trouver, autour du logiciel créé, des motivations autres que strictement techniques à l'implication. C'est notamment ce côté projectuel qui entraîne le refus, *a priori*, de plans de développement pré-établis, ainsi qu'une division des tâches par trop rigide. Le flou spatiotemporel des tâches à réaliser maintient l'incertitude sur le déroulement de l'activité et permet l'appropriation collective. Au-delà de la rationalité technique, ce sont des visions du monde et des imaginaires de transformation de la vie sociale que transportent avec eux les logiciels.

Diversité des activités et hiérarchisation des statuts

Les relations entretenues entre les membres du groupe initiateur comportent une composante interpersonnelle importante, qui permet de mettre à l'épreuve la force de l'accord sur l'identité du logiciel développé. Historiquement, les auteurs de Spip se rencontrent fréquemment dans le milieu associatif parisien, participent aux mêmes actions militantes, partagent des opinions similaires sur le développement de l'Internet et la liberté d'expression qui le caractérise.

De même, l'équipe initiale de développement de Claroline est confrontée concrètement à la problématique de l'usage des TIC à des fins d'enseignement, est associée à des activités de recherche sur ce sujet, partage quotidiennement un même espace de travail et de mêmes origines disciplinaires (science de l'éducation, philosophie, communication sociale).

Si le centre décisionnel est le garant et le défenseur de cette identité autour de laquelle il s'est constitué, il ne peut pour autant assurer la prise en charge de l'ensemble des activités liées à la production d'un logiciel. Car, au-delà du cœur informatique du logiciel, de multiples activités complémentaires sont nécessaires à sa diffusion : traduction, rapport de bogues, aide aux utilisateurs, ajout de fonctionnalités, mise en compatibilité avec des logiciels tiers, etc. L'enjeu communautaire consiste à susciter et maintenir un flux de contributions extérieures, enjeu d'autant plus incertain que celles-ci ne peuvent être prescrites ou commandées mais reposent uniquement sur le volon-

tariat de contributeurs distants. Si les ressorts de l'engagement de ceux-ci peuvent s'avérer très divers [Von Hippel, 2002] et inscrits dans des schèmes de carrières variées (Demazière, Horn et Jullien, 2004), la sensibilité aux fondements et à l'identité du projet s'avère généralement centrale dans la poursuite d'une implication soutenue. Mais cette condition n'est pas suffisante : pour que se développe une communauté, il est également nécessaire que, d'une part, une certaine répartition du travail soit organisée et que, d'autre part, se mettent en place des dispositifs de gestion afin de susciter l'intéressement des acteurs et une implication persistante.

Quelques traits saillants émergent des observations de terrain, sans conduire cependant à la formation d'un modèle unique. Tout d'abord, une division du travail s'instaure systématiquement et se cristallise dans une distribution stabilisée des tâches et même dans une différenciation de statuts hiérarchisés. Du point de vue des acteurs engagés, tous les participants sont membres de la communauté, ce qu'exprime le terme générique de "contributeur" pour désigner tout individu ayant participé à l'une quelconque de ces tâches. Mais l'éventail des tâches prises en charge exige un investissement d'inégale importance, pour certains très ponctuel (signaler un bogue), pour d'autres plus durable et intense (faire une documentation, maintenir une fonctionnalité). Les développeurs sont dispersés géographiquement, ont des activités professionnelles diverses, mobilisent des compétences différenciées, et ne bénéficient pas d'une reconnaissance identique au sein de la communauté concernée. C'est ainsi que la mention de l'identité du contributeur à certaines tâches, comme la contribution à l'écriture du code, introduit *de facto* des hiérarchies symboliques. L'usage de dispositifs de développement collectif (SVN, CVS, etc.), de gestion de listes de discussions ou de sites internes liés au projet nécessitent l'octroi de droits d'accès qui sont contingentés.

Émergent ainsi des différenciations entre contributeurs, qui distribuent des places, des attributions, des quasi statuts. Ici, la différenciation statutaire ne repose pas comme dans les organisations informatiques traditionnelles sur des compétences supposées que l'organisation apprécie le plus souvent à partir de critères indirects (diplôme, années d'expérience sur une technologie, etc.), mais elle traduit plutôt l'exhibition par la pratique de compétences mises en action et de l'engagement des participants [Tayon et Pitrou, 2005]. Ces statuts ne sont d'ailleurs pas inscrits dans des échelons hiérarchiques et n'alimentent pas des relations de subordination, mais ils stabilisent des positions dans des organisations de travail caractérisées souvent par un refus du formalisme.

Cette configuration dévoile, au minimum, une distinction entre un centre décisionnel et d'animation autour duquel gravitent des contributeurs périphériques, les communautés les plus développées ayant une structuration organisationnelle plus riche et plus complexe. Chaque apport des contributeurs périphériques est le plus souvent limité et s'inscrit dans le cadre fixé au projet dont il ne peut modifier profondément l'identité. Cependant, le fait que chaque contribution prise isolément est d'ampleur réduite ne doit pas conduire à sous-estimer l'impact de l'ensemble de ces contributions qui

p ermettent d'intégrer à moindre frais de multiples améliorations cumulatives permettant "d'exploiter au mieux une intelligence distribuée" [Foray, Zimmermann, 2001]. Autre caractéristique importante, entre contributeurs périphériques, comme avec les membres du centre, les relations sont principalement virtuelles, médiatisées par le réseau Internet. Les rencontres interpersonnelles sont rares et quelques fois non souhaitées : l'anonymat permet en effet de limiter son apport sans induire un engagement soutenu ou un dévoilement des ressorts de l'investissement personnel.

Des régulations nécessaires, une complexification croissante

À mesure que se développe cette couche de contributeurs distants se pose la question du ralliement d'acteurs aux motivations, comportements et attentes divers. Cette hétérogénéité croissante s'explique par les qualités institutionnelles différentes des acteurs qui s'investissent (simples individus, associations, entreprises, pouvoirs publics, etc.), les spécificités des compétences proposées (traduction, graphisme, documentation, codage, test), ou encore des opportunités qui apparaissent du fait de l'implication (lien de plus en plus étroit avec l'activité professionnelle, demandes de prestations de services, nécessité d'adaptation du logiciel à des besoins spécifiques).

Cette diversité nécessite d'être gérée dans l'optique à la fois d'attirer des apports, mais également de les canaliser dans des orientations contribuant au développement du projet et de sa visée normative. Mais il devient difficile pour les membres du centre de gérer directement de nombreux contributeurs. D'où l'apparition d'une multiplication des dispositifs techniques de gestion des contributions et des discussions, ainsi que d'un cercle supplémentaire constitué des personnes ayant un statut intermédiaire entre les membres du centre et les simples contributeurs périphériques. L'organisation interne des projets de logiciels libres peut être schématisée de manière typique en différents cercles délimités avec un degré variable de précision selon les cas et emboîtés les uns dans les autres.

Cercle stratégique	Liens interpersonnels forts, partage des mêmes orientations philosophiques et/ou dispositifs juridiques ou institutionnels exogènes
Cercle intermédiaire	Liens interpersonnels avec certains membres du groupe des fondateurs, prise de responsabilité dans l'animation de la (sous-)communauté
Cercle périphérique	Peu de liens interpersonnels : l'anonymat permet la collaboration d'individus aux motivations diverses, contributions limitées

La création de multiples listes de discussions spécialisées par objet témoigne de cette segmentation des communautés. Dans le cas Spip, des espaces de travail plus spécialisés se forment : division dans un premier temps de la liste principale en deux (Spip-dev et Spip-user), puis développement de listes plus spécialisées ; Spip-contrib pour la proposition d'ajouts de fonctionnalités optionnelles, Spip-trad pour la gestion des dizaines de traductions et le support aux non-francophones, Spip-lab pour la réflexion technicienne en termes de R&D, Spip-zone pour la stimulation de projets collaboratifs entre contributeurs, etc.

Chaque liste ou site a ainsi pour effet de rassembler des individus aux préoccupations communes et ciblées et de tenter de couvrir la diversité de centres d'intérêts qui soutiennent l'engagement dans le projet. Le fonctionnement de ces sous-espaces du projet est assuré par des participants particulièrement impliqués auxquels est attribué le statut "d'administrateurs". Ceux-ci ont la charge de gérer des dispositifs de communication (listes de discussions, sites Internet, serveurs collaboratifs) et d'animer les participants afin de susciter les contributions, de motiver l'implication, de s'assurer de l'entraide, mais également d'orienter, évaluer, sélectionner les apports. Ces couches intermédiaires bénéficient de l'approbation, voire d'un mandat du noyau stratégique qui se manifeste par la dispense de conseils, d'informations confidentielles, de contacts privilégiés.

En déléguant certaines activités et responsabilités, les membres du centre peuvent rétribuer par l'octroi d'un statut plus élevé les contributeurs périphériques les plus impliqués et donc soutenir la participation au projet. L'apparition d'un cercle intermédiaire indique également que les organisations ne sont pas figées, et signale que des passages d'un cercle à l'autre et donc des changements de statut sont possibles. Les promotions viennent sanctionner des activités importantes pour le projet. Pour cela, il est nécessaire que le contributeur ait fait la preuve de ses compétences techniques (appréciées à partir de la qualité de ses contributions) et de l'importance de son engagement dans la durée. Il faut aussi que les comportements du postulant soient en adéquation avec la philosophie du projet et montrent un accord profond avec l'identité du logiciel développé, et donc avec le système de valeurs qui sous-tend cette identité. Les cas de Claroline et de Spip mettent en évidence le caractère non automatique de l'intégration de code produit par des contributeurs distants du cœur du logiciel.

Ainsi G., développeur de Claroline relate le refus d'une proposition d'un étudiant grec : "L'entièreté de son code n'était pas exploitable tel quel, et il n'a jamais voulu accepter qu'on avait d'autres contraintes. C'étaient surtout des objections par rapport aux contradictions avec la philosophie de simplicité". Accorder les droits d'accès au développement du code source du logiciel – ce qui, à ce jour, ne s'est jamais produit – dépendrait fortement d'une implication soutenue : "Je crois qu'on est prêt à donner ces droits-là à quelqu'un à partir du moment où il y a déjà une certaine confiance qui s'est installée, que la personne est bien identifiée, donc de telle institution, que la

collaboration est réelle aussi. Il faut témoigner d'un peu de confiance, il faut témoigner du fait qu'ils ont déjà contribué et que ça s'est bien passé et que ça aide vraiment". Par conséquent, l'essentiel des contributions portent sur des activités de traduction ou de signalement de bugs : "Ce qu'on aime beaucoup, c'est "je ne propose pas de code, mais j'ai passé du temps à tester, tester". Ça, c'est à chaque fois du temps gagné" (C. Claroline). Les demandes de support sont redirigées par les développeurs vers le forum, "comme ça, avec un peu de chance, un utilisateur averti répondra avant nous" (H. Claroline).

Dans le cas de Spip, le contrôle centralisé des orientations du logiciel et des accès en écriture sur le cœur du logiciel en développement est souvent décrié. Du reste, les auteurs historiques se défendent d'un cloisonnement volontaire. L'accès aux droits de *commit* ne peut être envisagé, selon eux, ni comme un retour légitime récompensant une simple présence continue dans le projet, ni être restreint à un acte précis et temporaire.

Une fois accordés, ces droits ont la force d'un mandat, celui de poursuivre le développement en toute autonomie, et ne s'obtiennent qu'après avoir fait la preuve d'une grande qualité et fiabilité techniques : "Ce qu'apparemment personne ne comprend, c'est que l'on ne va pas ouvrir un accès comme étant une sucette ou un bon point, voilà. Ouvrir l'accès, ce sera à quelqu'un qui aura produit des *patchs* ou des trucs fiables, sérieux. On a du mal à vérifier ce que chacun de nous [les auteurs] fait, si en plus il faut corriger sans cesse le code par des gens qui sont pas au niveau, c'est pas la peine, c'est un travail épuisant. Un jour on en aura marre d'intégrer des patchs d'untel parce qu'untel il enverra toujours des patchs qui sont bons à intégrer puis, on en aura marre, et on leur dira tiens voilà, fais-le toi-même" (F., Spip).

Depuis le lancement du projet, deux contributeurs, au départ périphériques, ont ainsi rejoint le cercle des auteurs. Pour vérifier leur proximité idéologique au projet, les simples échanges virtuels communautaires sont apparus insuffisants ; ils ont été complétés par des interactions sociales directes avec des membres du centre (interaction en face à face ou échanges virtuels privés).

Cette exigence explique que des contributeurs qui ont pourtant effectué des contributions majeures (tant en termes d'importance que de difficulté) peuvent rester durablement à la périphérie de la communauté.

Différenciation, segmentation, et séparation

Il existe également des possibilités de régression dans la hiérarchie des statuts. Le problème se pose notamment quand un membre (du centre ou du cercle intermédiaire) n'effectue plus les tâches qui lui ont été déléguées. Faute de substrat contractuel aux relations et aux positions occupées, les possibilités de sanction par retrait des attributions correspondantes sont limitées et difficiles à mettre en œuvre. Plus fréquemment, la solution consiste à créer une nouvelle activité aux fonctionnalités proches de celle qui est mal réalisée, et à laisser ainsi dépérir cette dernière.

C'est le cas de B., initiateur de la liste Spip-contrib qui, lassé de la faible implication des personnes qu'il avait repérées et sollicitées à pratiquer l'évaluation de propositions de fonctionnalités complémentaires, abandonne ses fonctions et choisit de s'impliquer dans le développement d'un nouvel espace de travail (*Spip-zone*). Les modalités de fonctionnement de cette nouvelle institution doivent permettre d'améliorer le mode d'animation des contributeurs et de repenser le processus de sélection des autres administrateurs.

D'une certaine façon, cette pratique est assez proche de celle qui est à l'œuvre quand la légitimité du centre décisionnel est remise en cause par une partie de la communauté du logiciel concerné : plutôt que de tenter de destituer ces membres, la partie mécontente crée une nouvelle communauté avec un nouveau centre décisionnel à partir des développements déjà effectués (un *fork*), ce qui est rendu possible par les licences des logiciels libres (absence d'appropriation privée du code développé).

Cette probabilité est d'autant plus forte que le logiciel n'a pas encore connu un succès suffisant pour que la communauté ait dépassé une taille critique rendant peu crédible le développement d'un projet concurrent à partir du logiciel existant. Il existe ainsi dans le développement d'une communauté de production d'un logiciel libre une période délicate où, dans la terminologie de Hirschman (1972), le caractère peu coûteux de l' "exit" rend indispensable une gestion fine de la "voice" (notamment pour surmonter les incompréhensions réciproques) permettant de maintenir la "loyalty" et d'éviter les *forks*.

Les deux cas mobilisés ont tous deux fait l'objet de plusieurs stratégies de *fork*, les unes étant considérées plus "hostiles" que d'autres. Ici encore, la proximité des finalités poursuivies par des projets devenus concurrents détermine le caractère amical ou non de l'exercice et les déplacements ou segmentations des communautés entre elles. Spip fait l'objet depuis de nombreuses années de distributions concurrentes, dédiées plus spécifiquement au monde de l'éduction (Spipeva) ou intégrant des fonctionnalités supplémentaires (BioSpip). Mais en 2004, l'annonce d'une nouvelle version de Spip, développée par un prestataire informatique pour le compte d'une administration publique jette le trouble dans la communauté. Cette nouvelle version du logiciel (dénommée Spip-Agora) est présentée par le prestataire comme étant de meilleure qualité, comprenant des fonctionnalités supplémentaires, pouvant s'adapter à des environnements technologiques plus diversifiés.

L'intention déclarée du commanditaire est de faire bénéficier ces développements au projet initial afin de l'améliorer, de lui permettre d'élargir son marché potentiel aux administrations, et plus généralement, d'accroître sa crédibilité professionnelle. Mais le fait de n'avoir pas joué le jeu de la participation communautaire, de ne pas avoir distillé petit à petit les transformations réalisées, de ne pas avoir confronté les idées à la communauté et aux auteurs historiques en particulier, a provoqué un rejet de la part des auteurs.

Ceux-ci craignaient une récupération à des fins commerciales ou politiques du travail réalisé bénévolement par des dizaines de contributeurs, souvent

à partir de conditions de travail peu aisées, qui pourrait corrompre le caractère désintéressé de la communauté et provoquer une démotivation générale. Entre les deux projets qui coexistent désormais, les différences et les divergences s'approfondissent progressivement. Dans le cas de Claroline, la situation se présente différemment étant donné que c'est l'auteur historique du projet qui est dépossédé des rênes du développement par l'institution universitaire qui l'emploie. Ayant décroché un substantiel financement public pour parfaire le développement, il se voit, pour des raisons statutaires et politiques, refuser la direction du projet.

Il décide alors d'organiser son propre *fork*, en l'associant à la création d'une société de services. Ce départ brutal se concrétise sous la forme d'un e-mail envoyé à l'ensemble de la communauté, précisant notamment : "Quelqu'un a déposé la marque Claroline sans me demander mon avis et propose de me la revendre au prix fort. Suivant la philosophie de l'Open Source, j'ai décidé de ne pas accepter cela et de changer le nom du projet pour Dokeos. Veuillez noter qu'à partir de dorénavant, notre projet de logiciel de e-Learning Open Source ne s'appelle plus Claroline, mais Dokeos". Cette stratégie de communication lui permettra d'attirer avec lui une part importante de la communauté liée à Claroline.

Les communautés distantes

Ces deux cas de *fork* dits "hostiles" indiquent ainsi les facilités potentielles de transfert de contributeurs "périphériques" lors de situations de forks au travers de stratégies de communication. N'entretenant pas de relation spécifique entre eux en dehors des espaces publics de discussion, ni de relation interpersonnelles avec le cercle stratégique, ils ne peuvent décoder facilement les stratégies d'acteurs et les changements de cap dans l'identité des projets. Tout au plus peuvent-ils tenter d'évaluer les enjeux fonctionnels ou techniques des schismes, sans nécessairement percevoir par la transformation de la raison sociale du projet. Cette situation sera vécue différemment par les membres des cercles intermédiaires et stratégique qui, se réunissant ou entretenant des interactions privées, définiront des stratégies coordonnées de résistance et d'offensive afin d'éviter l'hémorragie communautaire.

Le *fork*, dans ce cas, peut exacerber des conflits interpersonnels et provoquer des inimitiés sociales qui poussent, par ailleurs, à l'accentuation des traits spécifiques de l'identité du projet. Les collectifs qui produisent des logiciels libres, et qui s'auto-désignent comme des communautés, présentent des caractéristiques peu répandues dans le monde du travail : les contributeurs sont dispersés et sont affranchis de toute contrainte issue d'une autorité extérieure au collectif constitué par la mise en réseau [Lerner et Tirole, 2002]. Comment une production est-elle possible dans ces conditions ?

On sait que distance relationnelle et identification à un groupe de projet ne sont pas contradictoires, et l'analyse des ressorts de la participation médiatisée par le réseau Internet a permis de renseigner ce paradoxe apparent, que

nous avons nommé ailleurs par l'expression "communautés distantes" [Demazière, Horn, Jullien, 2003]. Un autre aspect a été pris en compte ici, qui concerne la distribution des activités, l'organisation de la production et la structuration des relations de coopération. Car ces communautés distantes ont pour finalité le développement de logiciels informatiques opératoires, utilisables, performants. Sous réserve d'une généralisation des résultats, nous avons alors pu observer que ces activités à participation libre s'appuient sur une répartition distribuée des tâches et sur une hiérarchie informelle. Ces modes de structuration apparaissent comme des conditions pour le lancement, la poursuite et le développement d'activités orientées vers la fabrication d'un produit.

Au-delà, l'analyse des deux projets Claroline et Spip montre que pour être informelle, l'organisation n'en est pas moins visible à travers divers dispositifs : signature des contributions au code, identification des animateurs des listes de diffusion, marquage des statuts. Ainsi les relations entre les membres de la communauté s'insèrent dans des chemins et réseaux verticaux tout autant qu'horizontaux.

Ces groupes se caractérisent par un équilibre entre la cohésion et l'adhésion d'une part, la différenciation et la hiérarchie de l'autre ; entre d'un côté une identification, qui est modulée selon la place occupée dans le groupe et qui est fortement soutenue par l'identité discriminante du produit, et de l'autre une division, qui est assouplie par les mobilités internes, et qui est solidement légitimée par la visibilité des contributions individuelles. Ces collectifs sont ainsi constitués d'emboîtements de tâches, de statuts et de cercles, et sous cet aspect ils apparaissent comme des communautés non seulement distantes mais aussi, second paradoxe, clivées.

■

RÉFÉRENCES

D. DEMAZIÈRE, F. HORN, N. JULLIEN, "Le travail des développeurs de logiciels libres. La mobilisation dans des communautés distantes", Communication au colloque du CLERSE, La représentation économique de l'acteur au travail, Villeneuve d'Ascq, 20-21 novembre 2003, publiée dans CLES (*Cahiers Lillois d'Economie et de Sociologie*) n° 46, 2003.

D. FORAY, J.-B. ZIMMERMANN, "L'économie du logiciel libre : organisation coopérative et incitation à l'innovation", *Revue Economique 52*, pp. 77-93, 2001.

M. GENSOLLEN, "Biens informationnels et communautés médiatées", *Revue d'Economie Politique*, mars 2004.

R.L. GLASS, "A socio-political look at open source", Communications of the ACM, 46, pp. 21-23, 2003.

ALBERT O. HIRSCHMAN, *Face au déclin des entreprises et des institutions,* Éditions Ouvrières (Collection Economie humaine), 141 p., 1972, traduction de Exit, Voice and loyalty : responses to decline in firms, organizations and states, Harvard University Press, 1970.

HOLTGREWE U., WERLE R., "De-Commodifying Software ? Open Source Software Between Business Strategy and Social Movement", *Science Studies*, 15, pp. 43-65, 2001.

HORN F., *L'économie des logiciels*, La Découverte, Paris, 2004.

LEBRUN M., "Théories et méthodes pédagogiques pour enseigner et apprendre", De Boeck Université, Bruxelles, 2002.

LERNER J., TIROLE J., "Some simple economics of open source", *Journal of Industrial Economics*, 52, pp. 197-234, 2002.

RAYMOND E.S., "La cathédrale et le bazar, trad. de Blondeel S.", 1998, http://www.lifl.fr/~blondeel/traduc/Cathedral-bazaar/Main_file.html

SALAIS R., STORPER M., *Les mondes de production : enquête sur l'identité économique de la France*, éditions de l'EHESS, Paris, 1993.

TAYON J. ET PITROU A., "Libre Academy : statut ou compétence ?", Libroscope.org, 13 juin 2005.

VON HIPPEL, "Open Souce Software as horizontal innovation networks – by and for users", MIT Sloan School of Management W.P. N° 4366-02, 2002.