Les logiciels libres : une forme marchande ou une nouvelle hybridation organisationnelle

The open-source software: market form or new organizational hybridation

NEMMICHE Khadija
Maitre de conférences en sciences de gestion
C-U de Maghnia

BENDIABDELLAH Abdessalam
Professeur en sciences de gestion
Université de Tlemcen

Résumé :
Recourir au marché, à l'hiérarchie ou à l'externalisation, un sujet d'analyse largement débattu dans la littérature. Télécharger, acheter, internaliser ou même externaliser un logiciel se représente comme un dilemme organisationnel en formant un triplé analytique de la question de Sourcing. Cette substituabilité organisationnelle offre depuis plus d'une décennie une solution inter-commutable face aux nouveaux besoins applicatifs. Pour une nouvelle forme de logiciels appelée logiciel libre ou logiciel de source ouverte, la situation peut en quelque sorte être changée. Au lieu de recourir aux solutions marchandes habituellement choisies par les entreprises ou à internaliser un projet logiciel en se basant sur les services internes ou même confier ce projet à un prestataire externe spécialisé ; avec les logiciels libres on peut constater une nouvelle hybridation organisationnelle entre marché et hiérarchie qui ne revient ni aux relations strictement marchandes ni aux relations de subordination volontaire ni même aux relations hybrides de coopération interentreprises. Il s'agit d'une nouvelle forme via laquelle l'entreprise télécharge un logiciel dont l'accès à son code source est rendu possible. Cette démarche permet de restructurer ses fonctionnalités selon les besoins propres de l'entreprise. Ce papier vise donc à s'interroger sur cette nouvelle ère applicative dans le cadre de la problématique de Sourcing reconnue depuis longtemps afin de positionner structurellement les relations entre celui qui expose un logiciel libre dans le monde virtuel et celui qui le modifie en tant qu'une partie interne appartenant à une entreprise.
Mots clés : Logiciel Libre, Internalisation, Externalisation, Marché.
Abstract:
Resorting to Market, to Hierarchy or to Outsourcing is an analytical subject largely debated in literature. Downloading, Buying, Insourcing or Outsourcing software remains as an organizational dilemma forming an analytical triple of the question of sourcing. This organizational substitutability offers since more than one decade an inter-commutable solution facing new applications needs. For new software form's called free-software or open source software, the situation can be in a way changed. Instead of resorting to market solutions habitually chosen by firms or internalizing software project drawing on internal services or just as entrust an external specialized provider with this project; with free software we can observe a new hybrid organization between market and hierarchy that is not due to market relationships neither to voluntary subordination relationships' nor to hybrid relationships of inter-firms cooperation. It is about new form by witch a firm downloads software characterizing by ability to acceding to its source code. This step allows reorganizing its functionalities according to firm needs. So this paper aims to interrogate about this new application area in a form part of the problematic of sourcing known since a long time in order to place the relationships between the one proposing free software in the virtual world and the one modifying it as an internal party belonged to a firm.
Keywords: Open-Source, Insourcing, Outsourcing, Market.

Introduction :

Depuis l'article de Coase (1937) [1] la firme ou tout simplement l'entreprise dans le langage courant est devenue un objet d'étude pour plusieurs contributions scientifiques ayant pour mission

de mettre l'accent soit sur une question endogène soit sur un phénomène exogène qui relie la firme avec les différentes parties externes. La firme devient donc un centre d'analyse à travers lequel apparait une multiplicité de théories dont le but est de justifier les comportements internes notamment entre superviseur-subordonné, mais aussi externe qui visent à donner un cadre justifiant les décisions quand à l'environnement de la firme. Parmi celles-ci, la décision de Sourcing est l'une des principales problématiques qui ont pu occuper une place importante dans la littérature de la firme. Le choix entre internaliser ou acheter une fonction s'articulait autour de la question de la firme et ses frontières. Les chercheurs trouvaient dans cet angle un espace largement enrichi qui les permet d'approuver les différentes décisions qui ont été prises, de donner un plan de base pour les décisions qui sont en train d'être prises et de concevoir et prévoir les décisions qui pourront être choisies dans le futur.

Tardivement, émergea une nouvelle solution organisationnelle appelée forme hybride ou coopération interentreprises pour élargir le cadre applicatif de la décision de Sourcing. Cette nouvelle forme dont l'externalisation est un exemple a dilaté l'analyse en ce qui concerne la question de la firme et ses frontières. En conséquence, pour gérer une fonction, la firme se trouve jusqu'à nos jours face à un triplé inter-commutable qui la permet d'adopter soit la solution interne reconnue avec le concept internalisation, soit la solution hybride illustrée par les différentes négociations de long terme sous un cadre juridique et est reconnue par la notion de coopération interentreprises, ou encore externe présentée par des relations marchandes, simples et de très court terme.

Pour ce qui est de la fonction développement logiciel, une fonction récemment introduite dans la littérature économique et managériale, le chemin analytique de la question de Sourcing ne se change pas mais s'élargit en encastrant une nouvelle structure de relations interpersonnelles. Il s'agit d'une nouvelle solution qui appartient apparemment à la forme marchande, mais effectivement à une nouvelle figuration née d'une nouvelle hybridation organisationnelle. Ce sont les logiciels libres, les logiciels de code ou de source ouverte, également appelés logiciels open source. Ces logiciels qui avaient lieu dans le monde des affaires juste dans cette dernière décennie ont pu écraser la forme antécédente de logiciels à savoir les logiciels propriétaires.
La juvénilité de telle forme de logiciels ainsi que son applicabilité primitive qui revient principalement au domaine informatique laisse quelques lacunes extrêmement perçues qui doivent être rattrapées dans l'aspect économique et managérial de ceux-ci. Précisément dit, l'introduction très tardive des logiciels libres comme un objet d'étude dans les autres domaines de recherche a engendré un cadre partiellement abordé consacré à étudier cette forme de logiciels. Aujourd'hui les chercheurs en économie et en management mais aussi dans les autres domaines scientifiques hormis le domaine informatique généralisent leurs analyses quand aux logiciels libres en proposant une vision simpliste permettant de proposer une liaison de ces derniers avec un domaine de recherche particulier.

Cette superficialité abordée par exemple par les auteurs en économie et en management étant un résultat inévitable face à la récence de cette forme de logiciel a conduit à considérer ce dernier comme une forme marchande tel est le cas de logiciels propriétaires. Tandis que l'analyse de Sourcing permet en réalité et particulièrement pour les chercheurs qui se concentrent sur une telle question scientifique de retirer une nouvelle structure organisationnelle ou une nouvelle hybridation de solutions externes et internes. Cet aspect largement négligé a été présenté partiellement dans un seul travail mené par Demil et Lecock (2006)[2].
Le but de cet article sera donc de s'interroger sur l'originalité réelle de ce qui est dit par logiciels libres afin d'illustrer sa nature organisationnelle qui n'est pas en fait marchande comme est considéré souvent. Dès lors, cette tentative présentera une nouvelle hybridation organisationnelle

qui ne revient que partiellement à la structure marchande pour une forme particulière de logiciels à savoir les logiciels de source ouverte.

1- Le logiciel avec quatre mondes de production :

Développer un logiciel est défini avant tout comme un processus de création d'un produit logiciel, appréhendé comme une série d'étapes à réaliser et un ensemble de spécifications établies. En se référant à Salais et Storper (1993), Horn (2000[3], 2004[4]) identifie quatre mondes de production des logiciels. Ces quatre mondes proviennent du croisement de deux critères. Le premier est le degré de dédicace du produit logiciel via lequel on distingue des logiciels dédiés (également appelés spécifiques), et des logiciels génériques (ou anonymes). Le deuxième critère revient à la standardisation ou non- standardisation de méthodes et de composants utilisés dans la production des logiciels. L'interaction entre ces deux critères permet de distinguer entre: le monde de production flexible, le monde marchand (fordiste), le monde interpersonnel et le monde immatériel (ou de création) comme est illustré dans le tableau suivant :

Tableau (1) : Typologie des mondes de production

	Absence de standardisation	Standardisation du produit et/ou des composants
Produits dédiés	Monde interpersonnel	Monde de la production flexible
Produits génériques	Monde de la création	Monde fordiste

Source : d'après Horn (2000, p.443)[5]

Pour Horn (2004)[6], le monde interpersonnel repose sur la qualité spécialisé-dédié du produit. Plus spécifiquement, ce monde correspond à la production d'un logiciel sur mesure, développé en fonction de besoins spécifiques du client-utilisateur selon un processus spécialisé, et des compétences et savoirs propres à des firmes. Les principaux producteurs de ce type de logiciel sont les services informatiques internes de l'entreprise ou des SSII[7] le plus souvent spécialisées dans un domaine particulier quand l'activité est externalisée. Le monde de production interpersonnel reste un challenge majeur pour l'entreprise en général et pour la DSI en particulier car il présente un investissement lourd en termes de couts, de qualifications et du temps. Bien que la productivité est assez faible, mais ce monde de production répond typiquement aux besoins d'utilisateurs.

A l'opposé de ce premier monde se trouve le monde fordiste des progiciels commerciaux qui se caractérise par une production de masse répondant à des besoins standards. Les sociétés productrices sont appelées éditeurs de logiciels, et ce sont eux qui définissent unilatéralement les fonctionnalités de produits logiciels. L'essentiel de coûts réside dans le développement de l'original avec des coûts de reproduction extrêmement faibles. C'est pour cela que Horn (2004)[8] préfère d'y parler d'un fordisme spécifique vu que les coûts de production constituent presque exclusivement les coûts du développement de la copie originale. Ce monde de production se caractérise par des couts du développement faible dû à l'aptitude de producteurs à réaliser des économies d'échelle. Toutefois, la qualité de ces logiciels n'est pas jugée satisfaisante pour des produits destinés à remplir des besoins quasiment standards.

Le troisième monde de production est le monde de la création qui correspond au développement des logiciels libres. Ces derniers s'inscrivent dans la catégorie de produits génériques destinés aux utilisateurs anonymes, mais la principale particularité de ce monde de production et que ce sont les utilisateurs eux-mêmes qui réinterprètent les fonctionnalités génériques selon leurs exigences personnelles. La libre diffusion du code source du logiciel vise à atteindre une efficacité, une amélioration rapide du code et une suppression progressive des erreurs. De ce fait, certains logiciels libres ont atteint un très haut niveau de fiabilité et d'adaptation à

certains besoins de la communauté informatique, mais plus de difficultés pour répondre aux besoins de simples usagers.

Le dernier monde au sens de son apparition historique est le monde de production flexible. Celui-ci correspond au sur-mesure de masse qui prend deux formes : soit la fourniture autour d'un progiciel de services divers (installation, paramétrage, adaptation, formation, aide à l'utilisation, maintenance et évolution) ou le développement de logiciels sur mesure en réutilisant des modules logiciels déjà développés et testés, et en automatisant certaines phases du processus de production des logiciels (outils de génie logiciel). D'une autre manière, ce monde de production récemment introduit tente de dépasser les limites de la spécificité des produits logiciels d'une part et de la généralisation des progiciels d'autre part tout en essayant de standardiser le développement de fonctionnalités particulières ou d'offrir avec des produits génériques des services qui permettent de combler la lacune entre les besoins réels d'utilisateurs et les fonctionnalités réellement proposées en produits logiciels.

2- Du logiciel propriétaire au logiciel libre:

Un logiciel propriétaire[9] également dénommé logiciel non libre est un produit logiciel où l'éditeur offre uniquement l'usage du produit. Sa redistribution ou sa modification est donc interdite. Les logiciels propriétaires peuvent être classés en : logiciels commerciaux, partagiciels et gratuiciels.

Quand au logiciels commerciaux, Jadhav et Sonar (2009, p.556)[10] les définissent comme suivant: « Commercial software is software which is purchased through the retail market». Pour utiliser un logiciel commercial, le client doit faire l'achat. Les licences d'utilisation de ces logiciels interdisent tout accès au code source, toute sorte de duplication et de redistribution.

Cependant, un partagiciel encore appelé Shareware est un logiciel distribué gratuitement, éventuellement pour une période de test seulement et n'offre qu'un nombre limité de fonctionnalités. La rediffusion ou de modification d'un tel programme n'est pas autorisée. Après la période d'essai, l'utilisateur peut acheter la version complète du logiciel.

Tandis qu'un gratuiciel ou Freeware et comme son nom indique est offert gratuitement indépendamment de sa licence d'utilisation. Le code source du programme n'est pas disponible, ce qui interdit donc toute sorte de modification. Malgré que le gratuiciel est très proche du partagiciel, mais la principale différence réside dans le prix.

Contrairement au logiciel propriétaire, le logiciel libre également appelé Open Source divulgue librement le code source. Toute personne possédant un logiciel libre a le droit d'étudier, de copier, de divulguer et de modifier le code source. L'opinion fondamentale défendue par Richard Stallman le fondateur du logiciel libre et ses partisans est que la propriété intellectuelle et la protection des droits d'auteur ne concernent pas la création des logiciels. Les licences publiques confèrent ces quatre droits aux utilisateurs des logiciels libres, mais imposent à toute version modifiée d'être protégée par la même licence en interdisant les dépôts de licences classiques, c.-à-d. de rendre le logiciel libre un logiciel propriétaire (Lerner et Tirole, 2002)[11].

Reste à noter que cette forme de logiciels est qualifié libre parce que son accès au code-source est libre, sans aucune relation avec le prix (Gambardella et Hall, 2006)[12]. Néanmoins, traduire le mot free software engendre parfois un sens confus car pour les anglo-saxons free signifie libre, mais aussi gratuit. Tandis que le principe du logiciel libre n'a aucune relation avec le prix et la gratuité n'est pas donc un critère pour juger de la liberté du logiciel. Même si il existe des logiciels libres gratuits ou à bas prix par exemple via téléchargement, mais il faut comprendre que

la gratuité n'est pas une conséquence de la liberté du code source et de nombreuses sociétés commerciales sont d'ailleurs éditrices de logiciels libres[13] (Haefliger et al., 2008)[14].

Comparer les logiciels propriétaires avec ses compétiteurs stratégiques à savoir les logiciels libres indique une supériorité largement constatée de la dernière génération de logiciels au détriment de logiciels propriétaires. Selon plusieurs critères, il s'avère que le recours à un logiciel de source ouverte offre pour l'utilisateur plusieurs opportunités notamment en termes de qualité et de cout. Quand à la qualité, Bitzer (2004)[15] constate que celle-ci est supérieure dans un logiciel open source comparé avec les logiciels propriétaires en se référant à deux critères : le nombre des erreurs dans chacun de ces produits et la vitesse de résolution des problèmes. Certains auteurs considèrent ainsi que le logiciel libre est moins sujet au piratage puisque le code ouvert permet à plusieurs personnes d'identifier les défauts et de les corriger. Hertel et autres (2003)[16] mentionnent de leur part que les logiciels libres ont relativement moins de lignes de codes, ils peuvent ainsi fonctionner à partir d'ordinateurs moins puissants. De surcroit, et contrairement à un logiciel fermé qui ne répond pas toujours aux besoins d'utilisateurs, un logiciel open source permet aux développeurs d'éliminer des fonctionnalités non nécessaires et d'ajouter d'autres étant additionnelles selon leurs besoins. En outre, il est ainsi possible de remettre le logiciel libre dans le domaine public afin de favoriser des développements ultérieurs (arborescence du développement) (Lerner et Tirole, 2002)[17]. En effet, ce sont les utilisateurs eux-mêmes qui définissent les évolutions souhaitables, ce qu'un éditeur propriétaire ne pourrait réaliser qu'avec de nombreuses et coûteuses études du marché.

En termes de couts, les logiciels libres ont un coût faible comparés avec les logiciels privés. Même si certains logiciels propriétaires sont disponibles gratuitement, mais ceux qui ont un caractère lucratif sont plus chers par rapport à ceux qui s'inscrivent dans le cadre des logiciels Open source. En outre, Lerner et Tirole (2002)[18] soulignent que l'utilisation des logiciels de sources libres peut protéger l'entreprise contre les comportements de monopole et la surenchère des coûts des mises à jour. De plus, le fait que le client puisse évaluer le produit sans payer de licence est aussi un avantage en termes de diffusion. Toutefois, les logiciels propriétaires offrent un niveau de performance reconnu et un prix déterminé même si celui-ci se change en fonction de la politique commerciale. Ils fondent ainsi leur force sur une sécurité contractuelle et un service après vente.

3- Le logiciel avec trois mondes d'acquisition :

En tant qu'une décision stratégique, la question de faire ou de faire-faire a submergé ainsi la DSI[19]. Une concurrence de plus en plus forte dans une économie de l'immatériel, un environnement technologique turbulent qui engendre une forte éventualité de l'obsolescence technologique, une quantité informationnelle considérable venant à la fois de l'externe et de l'interne de l'entreprise rend du développement fréquent de projets logiciels comme une démarche inévitable pour la survie de l'entreprise. Après avoir considéré les logiciels comme un mécanisme indispensable dans la gestion quotidienne et stratégique interne et externe, la nécessité d'introduire de nouvelles fonctionnalités applicatives ou de modifier celles qui sont déjà mises en exploitation engendre une nécessité incessante à développer le logiciel. Face à ces exigences exhaustives, le choix d'un mode d'acquisition d'un logiciel reste toujours un fardeau, car la sélection d'une solution canonique entre internaliser, externaliser ou acheter un produit logiciel n'est pas une simple problématique à traiter pour la DSI. Cette dernière est obligée de faire un calcul approfondi et à mettre en balance les avantages et les inconvénients de chaque solution avant de se pencher sur l'une de ces dernières au détriment de deux autres (Brancheau et al., 1996)[20].

La première et la plus antique solution revient à internaliser le projet du développement logiciel en s'appuyant sur les services internes de la DSI. L'internalisation d'un logiciel est un

choix traditionnel depuis l'introduction des logiciels dans la vie des entreprises quand les entreprises n'ont pas eu une autre solution avant l'industrialisation des logiciels hormis l'Insourcing. Les autres solutions externes pour acquérir un logiciel n'avaient pas lieu quand les constructeurs proposent uniquement à leurs clients des systèmes propriétaires liés au système Hardware qu'ils produisent et où les SSII n'étaient pas encore nées avec une insuffisance marchande en offrant exclusivement des logiciels de base et langages de programmation (Genthon, Phan, 1999)[21]. Développer les logiciels par les informaticiens des organismes utilisateurs restait donc le seul mode d'acquisition des logiciels. Les entreprises constituent en cette époque leur stock d'applications spécifiques développées en interne en COBOL sur gros systèmes avec une productivité faible et des cycles de développement longs.

Ultérieurement, et après la conquête économique et sociale de TIC, les besoins d'entreprises en matières des logiciels en particulier ont fait naitre un nouveau marché appelé marché de progiciels dans lequel des sociétés spécialisées dans un segment particulier produisent et offrent en grande échelle des produits logiciels standards répondant aux besoins d'utilisateurs anonymes (Jiang et al 2001). Pour les entreprises utilisatrices, l'émergence de ce nouveau marché aux débuts des années quatre-vingt-dix permet de se défaire de l'insuffisance de ressources et moyens internes qui entravent pendant longtemps l'internalisation. Cela permet ainsi pour l'entreprise utilisatrice de ne pas perdre les ressources de la DSI dans des besoins standards facilement obtenus à moindre couts via le recours au marché. Il suffit de recourir au marché pour acquérir un progiciel en fonction de besoins d'utilisateurs. Dans ce sens, Welke (1981, p.400, cité par Mathiassen, 1998)[22] voit que : « it should be obvious that buying a package -- unless it is a poor fit or product -- will usually cost less money than creating it in-house. If you are not yet convinced of that statement, my suggestion would be to examine why you don't build your own automobile or make your own shoes».

Aujourd'hui, le marché de logiciels présente une solution non négligeable tant pour les entreprises que pour les autres parties de la société. Le recours aux progiciels qui reste la plus simple démarche occupe actuellement une place importante dans les choix de la DSI. Via un téléchargement par Internet ou l'achat de copies gravées sur un périphérique de stockage, l'entreprise accèdent à des milliers de produits applicatifs offerts à des prix attractifs. En termes de couts, les progiciels se classent en premier rang car la standardisation du processus de production et la simple duplication du programme permet aux éditeurs de réaliser des économies d'échelle importantes qui se reflètent immédiatement sur les prix de ventes de tels produits.

Tardivement ont émergé les SSII ayant pour mission de répondre aux besoins non standards vis-à-vis d'une inaptitude de la part du marché de progiciels à satisfaire certaines exigences de clients. Externaliser un projet logiciel aux prestataires spécialisés apparait pour plusieurs entreprises clientes une opportunité qui les rend capable de dépasser d'une part les challenges du développement interne et d'autre part la défaillance du marché externe (Bouchy, 1994)[23]. Les SSII se spécialisaient dans la programmation de logiciels spécifiques en s'appuyant sur des méthodologies très formalisées comme Merise et sur des outils logiciels. Leur clientèle est formée primitivement de très grandes entreprises publiques et privées. L'élargissement considérable de parts du marché occupées par ces SSII indique que les entreprises trouvent dans cette récente solution, un moyen par lequel elles peuvent confier une activité complexe comme le développement logiciel à une autre partie au lieu de sous ou de surinvestir dans des produits qui ne répondent pas parfois aux besoins demandés. Ainsi, l'externalisation d'un projet logiciel donne à l'entreprise cliente plus de convenance applicative aux besoins d'utilisateurs internes que ce n'est pas le cas de progiciels disponibles préalablement dans le marché.

La DSI joue donc un double rôle à la fois interne et externe dans un projet du développement logiciel. Le rôle interne de la DSI s'illustre dans le cas de l'internalisation du projet logiciel où la DSI est censée être le maitre d'œuvre ayant pour mission la production d'un logiciel selon les nouveaux besoins de la maitre d'ouvrage à savoir : les autres services internes de l'entreprise (Jadhav et Sonar, 2009)[24]. En revanche, le choix de l'une des solutions externes à savoir : l'externalisation ou le marché met de la DSI en situation de maitre d'ouvrage car c'est elle qui devient en contact direct avec les autres parties extérieurs représentant la maitre d'œuvre. Dès lors, la DSI joue un rôle de médiateur et de communicateur entre les services internes et les parties externes tout en assumant que le nouveau produit logiciel satisfera les besoins d'utilisateurs internes (Pinto et Slevin 1988)[25].

4- Une nouvelle forme de relations entre développeurs-utilisateurs :

Comme est cité plus haut la nouveauté de l'application de l'open source dans le monde des affaires entrave en quelque sorte de présenter la réalité organisationnelle de tel outil technologique. Le logiciel libre est souvent considéré comme un produit marchand ; un produit disponible par un simple téléchargement via internet. Quoique, la nature transactionnelle, contractuelle et fonctionnelle de ce dernier crée une particularité indiscutable quand à la question de Sourcing. Seuls Demil et Lecock (2006)[26] qui ont proposé un aspect de cette problématique négligée dans la littérature. Les auteurs voient que selon la TCT les projets applicatifs de sources ouvertes possèdent certaines caractéristiques qui les permettent de formuler une nouvelle structure indispensablement introduite dans le cadre de la question de la firme et ses frontières. Cette nouvelle forme organisationnelle prend selon les auteurs le concept de gouvernance du bazar. Les auteurs utilisent cette dernière notion en se référant à Eric Raymond le fondateur de logiciels libres en les qualifiant du monde bazar comparé avec les logiciels propriétaires qui reviennent de leur part au monde cathédral.

L'argument principal sur lequel se basent les auteurs en tentant de positionner structurellement et selon la supposition de la TCT les relations entre développeurs-utilisateurs des logiciels libres revient selon eux à la nature particulière de cette forme de relations qui ne reviennent ni aux relations strictement marchandes, ni aux relations de subordination volontaire ni même aux relations de coopération interentreprises. La nature de contrats d'utilisation de quatre libertés rendues possible par le choix d'un projet de source ouverte mais aussi la nature du contrôle durant le projet du développement d'un logiciel open source crée une nouvelle structure qui doit être soulignée et se présenter comme une nouvelle forme organisationnelle à coté de la trilogie formant le choix de Sourcing.

La liberté donnée aux utilisateurs pour accéder au code-source et le modifier, le copier et le partager a crée une nouvelle forme de coopération particulière entre ceux qui considèrent le logiciel comme un produit et ceux qui le considèrent comme un outil. De cet angle, on peut constater deux sortes d'utilisateurs qui présentent deux natures de relations sur lesquelles nous basons notre supposition.

Tout d'abord, il est fort de dire que la réussite extraordinaire de l'open source trouve ses origines dans un groupement de personnes appelé communauté du logiciel libre. Cette dernière forme englobe des relations coopératives via lesquelles leurs membres partagent des connaissances, des expériences, mais aussi des produits conjointement développés à savoir : les logiciels libres. Ces membres sont géographiquement dispersés, mais technologiquement reliées à travers l'Internet. En effet, l'Internet est tout simplement la raison d'être de la communauté de logiciels libres. Mais il faut également citer que n'existe aucune clause dans les licences open source qui impose la coopération avec les autres dans la production et le développement d'un programme libre ou plus généralement qui définit la manière dont on doit fabriquer le code. De ce fait, produire un

logiciel libre peut être un travail solitaire. Un programmeur rédige individuellement son code-source, le compile, le teste et le met sous licence GPL à la disposition de tous. Cette situation est très fréquente et prend la dénomination du modèle de la cave. La communauté du logiciel libre présente le noyau de l'analyse menée par Demil et Lecock (2006)[27] qui ont pour ambition d'étudier cette forme de relations coopérative en tant qu'une nouvelle structure de gouvernance.

Deuxièmement, il n'est pas indispensable que celui qui utilise le logiciel libre devient un membre d'une communauté open source, mais toute partie possède le droit d'utiliser les fonctionnalités du logiciel libre sans même d'avoir besoin à accéder au code source. Néanmoins, on doit exclure ce dernier cas de notre analyse car une simple utilisation du logiciel libre sans profiter de quatre libertés données par l'acquisition d'un tel produit lui rend un produit purement marchand. Dès lors notre analyse s'appuie de ce deuxième angle sur l'utilisateur qui en téléchargeant le logiciel libre tente de mener des modifications selon leurs exigences propres et ce lui permet de substituer les autres solutions organisationnelles à travers lesquelles il acquiert un nouveau logiciel. Précisément dit, on y parle d'une firme cliente qui face à l'émergence de nouveaux besoins exprimés en la seine peut répondre à ceux-ci en changeant une partie de fonctionnalités déjà disponibles dans un logiciel libre pré-packagé.

Considérer la relation entre développeurs-utilisateurs comme une forme organisationnelle particulière qui ne revient ni à la hiérarchie ni au marché ni à l'externalisation provient de la nature de relations entre ces deux principaux pôles et de la manière selon laquelle ils gèrent les projets de source ouverte. Selon la définition Williamsonnienne de la transaction: « Transactions occur whenever 'a good or service is transferred across a technologically separable interface. One stage of processing or assembly activity terminates and another begins" (Williamson, 1981, p.1544)[28], les deux parties de notre analyse à savoir : les développeurs et les utilisateurs -formant une communauté de logiciels libres ou non- sont géographiquement dispersés en échangeant un produit disponible via téléchargement qui est le logiciel open source. En conséquence, une transaction se représente entre deux acteurs indépendants. En outre, cette transaction prend différentes formes entre utilisateurs-développeurs selon le but de la partie utilisatrice. Cette dernière peut tester et corriger les bogues applicatifs ou juste donner leur rapport sur le produit logiciel en s'inscrivant selon O'Mahony (2003) en ce que nous appelons assurances qualité. Dans d'autres cas, l'utilisateur conçoit ou mène des nouvelles bogues à un logiciel donné en y parlant de l'innovation (Lee et Cole, 2003). De plus, et comme dans la communauté de AHU[29] les membres organisent un service dont lequel ils proposent des réponses à des questions posées par le groupe et ce permet d'améliorer l'espace du développement logiciel.

Dès lors, les relations entre des parties autonomes prennent différentes formes discriminantes qui ne reposent sur aucune analogie avec les autres formes de relations organisationnelles formant la triplé organisationnelle reconnue depuis longtemps.

Dans une autre part, les licences d'utilisation de l'open source assurent qu'aucune partie ne devient propriétaire de fonctionnalités originales du produit. La Licence Publique Generale GPL, GNU et Berkeley Public Licence qui sont des exemples de licences publiques confèrent les quatre droits aux utilisateurs des logiciels libres, mais imposent à toute version modifiée par ceux-ci d'être protégée par la même licence en interdisant les dépôts de licences classiques. Autrement dit, ces licences empêchent de rendre le logiciel libre un logiciel propriétaire (Lerner et Tirole, 2002)[30].

En effet, la gouvernance du Bazar mérite d'être considérée comme une institution particulière alternative à la hiérarchie où la firme confie le projet du développement à leur services internes, au marché où la firme télécharge ou achète un progiciel et à l'externalisation où la firme confie le projet du développement à un prestataire spécialisé. Cette dernière forme de relations

plurales se représente quand la partie utilisatrice qui revient dans cette analyse à la firme télécharge un logiciel libre, accède à son code source et corrige ou mène certaines modifications fonctionnelles pour le rendre usable en répondant à des nouveaux besoins exprimés à l'intérieur de la firme.

5- Caractéristiques de la nouvelle structure de gouvernance :

Pour retirer les caractéristiques de la gouvernance du bazar, il faut mettre l'accent sur les différences catégoriques entre cette nouvelle institution et les trois autres structures de gouvernance à savoir : la hiérarchie, le marché et l'externalisation.

Quand à la hiérarchie, la gouvernance du bazar se caractérise par des particularités très distinctives qui rendent celle-ci complètement dissemblable de la structure interne. Quand aux relations bilatérales au sein d'une firme, celles-ci sont verticalement structurées en présentant un système autoritaire et une subordination volontaire telle est soulignée par Williamson (1975)[31]. Au contraire, cette nouvelle hybridation organisationnelle repose sur des relations bilatéralement indépendantes où aucune partie n'est obligée d'exécuter certaines taches. Les relations entre utilisateurs-développeurs sous un cadre d'une gouvernance du bazar sont donc horizontalement structurées et l'exécution des devoirs au profit du propriétaire n'est pas garantie conduisant parfois à interrompre le projet du développement s'il n'exista pas des participants volontaristes dans une communauté donnée.

Pour ce qui est du contrat du travail qui reste toujours le seul instrument qui assure la continuation d'une relation de travail et l'acceptation d'une subordination unilatérale au sein de la firme étant considérée comme un nœud de contrat comme est indiqué par Jensen et Meckeling (1976)[32], dans une gouvernance du bazar l'absence du contrat permet d'élargir les formes selon lesquelles les relations entre les deux parties s'exécutent. Ces relations se caractérisent par une indépendance totale assimilée à celle qui se trouve dans une transaction marchande où l'autonomie décisionnelle s'impose. Ainsi, les licences publiques de logiciels libres rétrécissent en quelque sorte la liberté donnée aux utilisateurs de ces formes de logiciels comme est cité précédemment. Dès lors, si l'éventualité de se comporter d'une manière opportuniste dans une hiérarchie est cernée via une passation d'un contrat de travail, au sein d'une gouvernance de bazar, où les procédures de contrôle et de surveillance ayant pour objectif de garantir les droits de chaque partie sont inexistants, les licences d'utilisation de l'open source qui donne plus de liberté à l'utilisateur entrave en même temps toute comportement illégal sous forme d'un profit crée par une propriétarisation d'un logiciel libre.

D'une autre part, l'utilisation des logiciels libres permet à la firme cliente de rétrécir le processus du développement. Au lieu de perdre une longue durée dans l'étude des besoins, de l'exécution du projet et dans le test et la vérification, par les logiciels libres, le temps durant lequel le projet sera disponible est très court car les utilisateurs internes ajoutent quelques modifications généralement périphériques pour que le produit soit disponible à être exécuté.

La différence entre la gouvernance du bazar et le marché est ainsi claire. Pour une transaction marchande, il est nécessaire d'identifier le vendeur. En achetant ou en téléchargeant un logiciel, l'utilisateur doit connaitre l'éditeur de ce dernier et ce permet de réduire en quelque sorte l'incertitude qui entoure un produit privé dont l'accès à son code source est impossible. Quand aux parties formant la nouvelle structure de gouvernance qualifiée de bazar, l'identification de celles-ci n'est pas obligatoire. L'aptitude à accéder au code source par l'utilisateur lui permet de ne s'intéresser qu'aux fonctionnalités existantes dans le logiciel et non pas à la partie développeuse, car l'utilisateur est capable de restructurer ces fonctionnalités et les tester tout seul et indépendamment du développeur. En conséquence, l'incertitude qui entoure la fiabilité du produit

en obligeant l'identification des parties est restreinte (Demil et Lecock, 2006)[33]. Ainsi, l'absence d'un contact direct entre la firme cliente et l'éditeur du logiciel engendre des problèmes inévitables. Quand la firme achète un progiciel fonctionnel, sectoriel ou trans-sectoriel et trouve des embarras de sa mise en exploitation, elle est obligée de rejeter le produit, d'attendre une nouvelle version qui dans le cas échéant comportera des corrections de bogues ou d'adopter une autre solution organisationnelle. En choisissant un logiciel libre, la firme cliente peut suivre des modifications immédiates une fois que ces embarras s'imposent.

Comparé à l'externalisation, la gouvernance du bazar est extrêmement dissemblable. Si la première reposa sur une relation d'une durée moyenne ou longue, la relation dans une gouvernance du bazar est très courte et se termine le temps où le téléchargement du logiciel libre se termine. Dans le cas contraire, la relation d'externalisation qui implique l'intervention de deux pôles économiques juridiquement indépendants se prolonge jusqu'à une période prédéterminé dans un contrat passé par les deux acteurs selon lequel l'identification de ces derniers est l'une des clauses indispensables pour que cette dernière puisse avoir lieu. De ce fait, la longue durée du contrat fait naitre un problème d'incertitude inévitable qui oblige les parties contractantes à redéfinir les clauses du contrat via une renégociation parfois récurrente ayant pour but d'inclure les nouveaux événements ou besoins applicatifs. Cependant, le recours aux logiciels libres se particularise par un dynamisme autonome où la firme cliente intègre toute sorte de changement imprévu indépendamment du développeur original. De plus, après la réinternalisation d'un nouveau logiciel externalisé à un tiers, la firme cliente devient dépendante de son prestataire. En cas de détection des erreurs dans le logiciel, de l'appariation des problèmes de mise en exploitation ou de non adéquation au besoins, la firme cliente sera obligée de recourir récursivement à son prestataire. En revanche, ces problèmes seront immédiatement réglés par la firme cliente toute seule si elle choisit la gouvernance du bazar. L'accès au code source du logiciel permet aux utilisateurs internes de restructurer ses fonctionnalités en corrigeant indépendamment tout problème éventuel.

Conclusion :

Cette tentative pionnière avait pour but d'instaurer théoriquement une nouvelle institution négligée jusqu'à nos jours par les chercheurs. En se concentrant sur la question de Sourcing qui occupe une place importante dans la littérature managériale, on a pu dans ce travail présenter une nouvelle hybridation organisationnelle née d'une combinaison de nouvelles formes de relations entre des parties juridiquement indépendantes. La firme qui était pendant longtemps face à trois solutions traditionnelles en vue de gérer une fonction se trouve aujourd'hui pour une fonction particulière face à une quatrième solution indiscutable qui permet d'élargir les choix de la firme en offrant plus de convenance et de rationalisation à la démarche choisie.

Les nouveaux besoins applicatifs exprimés par les utilisateurs internes de la firme qui deviennent fonctionnellement de plus en plus compliqués et sophistiqués complexifient les processus décisionnels de la firme pour ce qui est du sujet de la firme et ses frontières. La firme peut internaliser un projet logiciel en le confiant à ses services internes spécialisés, elle peut encore externaliser ce projet à un prestataire externe en externalisant les besoins applicatifs internes, mais elle peut aussi recourir au marché en achetant ou en téléchargeant des produits logiciels destinés à remplir des besoins standards d'utilisateurs anonymes. La quatrième solution sera donc le choix des logiciels libres qui crée une nouvelle forme organisationnelle appelée gouvernance du bazar. Cette dernière institution conçue dans ce papier se représente quand la firme télécharge un logiciel dont l'accès à son code source c.-à-d. à ces fonctionnalités algorithmiques est faisable par les utilisateurs internes. Cette faisabilité les permet de mener des corrections ou des modifications immédiates sans supporter les conséquences négatives du choix.

Notre supposition dans ce travail s'appuie sur une contribution précédente menée par Demil et Lecock dans un travail publié en 2006[34]. Dans leur étude, les auteurs proposent pour la première fois que la communauté des logiciels libres mérite d'être considérée comme une nouvelle forme organisationnelle différente de trois autres structures de gouvernance encadrant la problématique de Sourcing. Notre touche revient à élargir cette supposition en présentant une conception plus étalée de relations entre développeurs-utilisateurs autrement dit entre la bipolarisation de logiciels libres. Notre conclusion ouvre donc une nouvelle porte pour illustrer l'aspect économique et managérial du logiciel libre où on a bien retiré la différence entre ce qui est dit par gouvernance du bazar et le marché, la hiérarchie et l'externalisation. Cette nouvelle forme de relations dessine une solution plus appropriée pour la firme cliente mais aussi un nouveau chemin plus fécond pour que les chercheurs puissent se concentrer sur ce nouveau cadre de recherche. En ce travail, on a bien illustré la place affranchie de logiciels libres qui restent depuis son introduction dans le cadre économique une forme faisant partie de produits marchands. Les traces que laissent les particularités fonctionnelles mais aussi relationnelles de cette nouvelle génération de logiciels exposent clairement sa dissimilitude en faisant naitre récemment une autre ramification subsidiaire de la problématique de Sourcing née par Coase depuis prés d'un siècle.

Notes et références bibliographiques :

[1]- Coase, R. H. (1937): The nature of the firm, Economica, Vol.4, pp.386-405.
[2]- Demil B., Lecocq X., (2006): Neither Market nor Hierarchy nor Network: The Emergence of Bazaar Governance.Organization Studies, Vol. 27 Issue 10, pp.1447-1466.
[3]- Horn F, (2000) : De l'économie de l'informatique à l'économie du logiciel, thèse de doctorat en économie industrielle, Université de Lille.
[4]- Horn F., (2004) : L'économie du logiciel, La Documentation Française.
[5]- IBID;
[6]- IBID;
[7]- Sociétés de services et d'ingénierie informatique.
[8]- IBID;
[9]- Même si la plupart des logiciels commerciaux sont propriétaires, mais commercial et propriétaire ne sont pas des synonymes car il existe des logiciels propriétaires gratuits.
[10]- Jadhav A.S, Sonar R.M., (2009): Evaluating and selecting software packages: A review, Information and Software Technology, Vol.51, pp.555-563.
[11]- Lerner, J., Tirole, J. (2002): Some Simple Economics of Open Source, *Journal of Industrial Economics*, Vol.50, pp.197–234.
[12]- Gambardella A., Hall B. H. (2006): Proprietary versus public domain licensing of software and research products, Research Policy, Vol. 35, n.6, pp.875-892.
[13]- En réalité ces sociétés commercialisent des activités liées au logiciel (sélection de logiciels, réalisation de la copie, distribution, garantie, maintenance, intégration, conseil, installation, assistance technique, développement de solutions spécifiques…), et non pas le logiciel lui-même : une personne intéressée seulement par le logiciel, peut le télécharger gratuitement et le compiler à partir de son code-source librement accessible.
[14]- Haefliger S., Von Krogh G., Spaeth S. (2008): Code reuse in open source software, Management Science, Vol. 54, n.1, pp.180-193.
[15]- Bitzer, J., (2004): Commercial versus open source software: The role of product heterogeneity in competition, Economic Systems, Vol.28, pp.369-381.
[16]- Hertel G., Niedner S., Herrmann S., (2003): Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux kernel, *Research Policy*, Vol. 32, n.7, pp.1159-1177.
[17]- IBID;
[18]- IBID;
[19]- Direction des systèmes d'information : étant la seule partie responsable du développement, de l'exploitation et de la gestion des systèmes d'information de l'entreprise.

[20]- Brancheau J.C., Janz B.D., Wetherbe, J.C. (1996): Key issues in information systems management: 1994-1995 SIM Delphi results, MIS Quarterly, Vol.20, n.2, pp. 225-242.
[21]- Genthon C., Fan D. (1999) : Les logiciels libres : un nouveau modèle ?, *Revue Terminal*, Technologie de l'information et société, N80/81, pp.167-188.
[22]- Mathiassen L. (1998): Reflective systems development, *Scandinavian journal oh Information Systems*, Vol.10, n.2, pp.67-118.
[23]- Bouchy S. (1994) : l'ingénierie des systèmes d'information évolutifs, Eyrolles.
[24]- IBID;
[25]- Pinto, J. K., Prescott, J. E. (1988): Variations in Critical Success Factors Over the Stages in the Project Life Lerner J., Tirole J., (2000): The Simple Economics of Open Source, NBER Working Paper Series 7600.
[26]- IBID;
[27]- IBID;
[28]- Williamson, O. (1981): The Economics of Organization: The Transaction Cost Approach, American Journal of Sociology, Vol.87, pp. 1540-1567.
[29]- Apache Help Usenet.
[30]- IBID;
[31]- Williamson O.E. (1975): Markets and Hierarchies: Analysis and Antitrust Implications.New York, The Free Press.
[32]- Jensen M., Meckling C., (1976): Theory of the Firm: Managerial Behavior, Agency Costs and Ownership Structure, J Fin Econ, pp.305-360.
[33]- IBID;
[34]- IBID;