

***The Role of Free Licences for Cooperation Development and Learning  
within the Commons-Based Peer Production.  
The Cases of CC and GNU GPL.***

*Inna Lyubareva<sup>1</sup>, Barbara Feledziak<sup>2</sup>*

**Abstract.**

Development and expansion of the Commons-based peer production in different spheres is of special attention for economists for many reasons. One of these reasons is the problem of cooperation development among individuals dispersed across space, time, and organizational boundaries. Our starting point is that to create collectively a product goes far beyond the sharing of distinct ideas, but necessitates complex processes of individual and collective learning to ensure the compatibility of emergent rules, norms and routines. In this paper we will discuss how free licences can contribute to the co-ordination of distributed pieces of knowledge and, hence, to the coherence of the production process.

**Acknowledgements.**

We gratefully acknowledge the helpful comments and suggestions of Eric Brousseau, Michel Delapierre, Frederic Gannon, Fabrice Rochelandet and Maria Alessandra Rossi, as well as the permission to use the study on the Open Source Software realized in July 2005 by Nicolas Auray and Michael Vicente, ENST, Paris. We also thank the leader of Take the Bus project, Philippe Chavaroché (Ana Boat), and the creator of the Musique-libre.org platform, Bituur Esztrem, for informative discussions. None of those who have helped can be held responsible for defects that have remained, or for the views expressed here.

---

<sup>1</sup> Paris X – EconomiX; inna.liubareva@u-paris10.fr

<sup>2</sup> Paris X – EconomiX; barbara.feledziak@u-paris10.fr

## **Introduction.**

The emergence and expansion of a specific mode of production also called the commons-based peer production (CBPP<sup>3</sup>) is a recent phenomenon. The most well-known case where the CBPP is applied is development of the open source software (OSS). However, its range of application has expanded quickly to other domains such as music creation and literature. CBPP is neither centrally planned nor profit driven: production is organized through the free cooperation (not contract-based<sup>4</sup>) of agents dispersed across time and space and engaged in the production of common resources, without relying on hierarchical methods of command and control. From this perspective the CBPP can be presented as a decentralized and distributed system, allowing to take important advantages from a potential of distributed knowledge [Jullien & Zimmermann, 06]. However, by connecting a large amount of individuals working on a common project in such a decentralized and distributed manner, this mode of production could make it more difficult to ensure the development of cooperation within projects implying complex processes of individual and collective learning: some specific coordination mechanisms are necessary “to ensure at the same time decentralisation and coherence of the project advancement” [ib.]. So, CBPP is of special interest for economists for many reasons, and one of them is that its coordination efficiency is ensured through an alternative manner than in a formal organization: collective production consistency is ensured while preserving high flexibility and wide extent of resources.

In this paper we deal with the problem of coordination of the CBPP and propose to consider the role of free licences for the coherence of the collective creation<sup>5</sup>. The essence of these licences consists in the idea that “authors do not renounce to their rights but only to the

---

<sup>3</sup> The concept was firstly defined as “Commons-based peer production” by Y. Benckler in “Coase’s Penguin, or Linux and the Nature of the Firm”.

<sup>4</sup> Despite the fact that some participants in the CBPP can be paid to contribute full-time to the projects, the majority of members are volunteers. This is in particular the case in the projects that we propose to consider in the paper.

<sup>5</sup> Usually economic literature on the IPRs considers licences (and other forms of intellectual property protection) as incentive mechanism that guides the allocation of resources for the generation of new knowledge allowing the control over the product and the appropriation of economic returns. It considers PRs as necessary condition for the establishment of cooperation, but often it doesn’t take into account that there may be a process between the establishment of appropriate integrating mechanism and the achievement of efficient cooperation. The question of the role of PRs *within* the process of knowledge creation is our special interest. However this problem surpasses the framework of this study. So the current paper is just the first step towards the analysis of this problematic: for the moment we propose to overlook the incentive aspect of free licences and, hence, to put aside their role as a specific form of intellectual property.

monopoly rent such rights would authorise in a copyright regime” [ib.]. In practice it means that functional work (such as software, encyclopaedias or dictionaries), artwork or other creative contents are publicly available (for example, on the Internet) and can be shared among users. We argue that free licences (as *one of* the coordination mechanisms) enable not only sharing of distinct ideas but also projects’ coherence due to the eventual compatibility of emergent rules, norms and routines through which high quality of products is achieved. We will show also that the way the coherence within the creation process is achieved (the structure of the licences) depends on the particular characteristics of the creative content.

Our analysis relies on the comparative study of the cases of collective music and software creation (the Take the Bus project and the Debian project respectively) and aims at explaining how individuals dispersed across space, time, and organizational boundaries co-ordinate their distributed contributions to create collectively a product.

## **Methodology.**

For our purpose, we adopt a knowledge-based view of the organization. It means that we put aside the question of the individual agent’s motivations to cooperate, to focus on the process of cooperation development, involving dynamic individual and collective learning. We suppose that the development of cooperation results in the eventual compatibility of emerging rules, norms and routines. In other words, the dynamic process of individual and collective learning provides the basis for the coherence of production. “Coherence refers to the co-ordination of distributed pieces of knowledge and involves the creation of commonly shared bodies of knowledge: sets of facts, notions, "models of the world", organizational routines, rules, procedures which are – at least partly – known to all the members of the organization involved in a given interaction” [Cohendet & Llerena, 01].

Knowledge-based view of organization supposes the two-sided approach to the question of production coherence.

On the one side, the body of literature [Kogut & Zander, 92; Hodgson, 98; Osterloh & Frey, 00] emphasizes the crucial role of an organization (a Firm) both for the cooperation development and the coherence of production at least for two reasons: firstly, because a single

institutional framework provides a “protected cultural enclave” [Hodgson, 98; Johnson, 92; Daft & Weick, 84] where collective and individual learning take place (because they are context-dependant and culture-bound); and secondly, because hierarchy, while it cannot have strong influences on the internal dynamic evolution of cooperation, can however stimulate the structuring of cooperation among individuals and groups of individuals [Amin & Cohendet, 04].

On the other side, the evolutionary literature on production coherence [Dosi and al., 92] emphasizes that the way the coherence is achieved depends on the characteristics of production processes, because “learning is production-specific” [Hodgson, 98].

Therefore, our question is: how the coherence of production, in terms of its consistency, is achieved within decentralised and distributed systems of CBPP? We suppose that free licences like the General Public Licence (GPL) and the Creative Commons licence (CC)<sup>6</sup>, which protect knowledge sharing, play also a specific role for the development of cooperation.

Hence, in order to present the free licences as a mechanism stimulating individual and collective learning in the case of CBPP we should take into account two main aspects:

- They must stimulate the development of cooperation (individual and collective learning) in the framework of the decentralised and distributed organization of the CBPP;
- They must meet the specific characteristics of the process of product creation.

### **Structure of the paper.**

In the first section we will describe briefly the history of the CBPP evolution and of the development of free licences. In the second section we will show that CBPP provides some important advantages, in terms of product quality, for the development of creative content and that these advantages stem from the decentralized and distributed nature of production organization. In the third section we will analyse how in the cases of the Debian project (the GPL licence) and the Take the Bus project (the Creative Commons licence) the free licences contribute to the creation of “shared context” and common culture. Finally in the fourth section we will

---

<sup>6</sup> The list of the free licences is much larger than the GPL or the CC licence and includes also, for example the BSD licence which is often used in the Open Source Software. However the GPL and CC are to a certain extent emblematic for the CBPP in the spheres of software and music development which we analyse in this paper.

discuss how these licences meet the specificities of production processes in both cases; we will study the processes of music and software creation according to three characteristics: complementarity of assets, convergence of contributions and learning aspects.

## **1. Evolution of the CBPP and Creation of Free Licences.**

The history of the CBPP begins with the development of the Unix operating system when, at the very beginning of the 80s, AT&T Bell Labs launched the software with the source code in order to get a large number of software developers to take part in the development of Unix. Many developers were interested to participate in its development; however the success of the project was less than expected. The main reason of this relative failure was that during the 1980s Unix split into a variety of different (free and proprietary) competing versions which were incompatible. There is a viewpoint that notably this product differentiation did great damage to the Unix market: “The Unix wars were the struggles between vendors of the Unix computer operating system in the late 1980s and early 1990s to set the standard for Unix henceforth. These battles are commonly held to have harmed the market acceptance of Unix and created a market gap that allowed the rise of Microsoft Windows NT.”<sup>7</sup>

The creation in 1984 of the first free licence, GNU General Public Licence (GPL), by Richard Stallman was a response to the inefficiencies faced by the Unix project. GPL mobilises in a specific manner the similar principles as traditional copyright, but in order to protect four essential users’ rights: freedom to run the program, to study how it works, to modify the program, and to redistribute it. This licence is based both on the copyleft principle and on the paternity principle. Copyleft corresponds to the restriction that all new pieces of code, “derived work”, must be licensed under the GPL and, hence, the source code must be available to all users of the software. The paternity principle means that all the contributors participating to the software creation have to be named in the derived work. Later, different licences have been created within the OSS (for instance, BSD), however the GPL remains the most widely used licence (today about 70% of all the projects are published under the GPL).

---

<sup>7</sup> [http://en.wikipedia.org/wiki/UNIX\\_wars](http://en.wikipedia.org/wiki/UNIX_wars)

With the implementation of the GPL the CBPP has begun to expand rapidly in the software domain. The success of the OSS movement can be ascertained by the fact that numerous projects exist today, with many of them competing with their proprietary analogues. Among the most emblematic projects are Apache, Linux and Debian.

Furthermore, the success of the CBPP in the software “industry” has attracted attention to the CBPP mode from other spheres. For example, in 2000, actors from the OSS movement and musicians have organized a meeting called “the Copyleft Attitude” which has resulted in the creation of the first free licence for the music content, the Art Free License. However this licence which was analogous to the GPL was not as successful and in 2001 the Creative Commons licence (CC) settled in the area of music creation. Compared to the Art Free License, the CC gives the possibility to the authors to control some use of their works and to keep their status of artists: authors have the possibility to choose from an array of options, in particular to permit or not the commercial use or derivative works, as well as to require or not attribution and share alike. The first American version of CC was issued in 2002, while in France the final version was ready in 2004 and gained an enormous success. Despite the fact that the CCPB among the musicians is relatively new, today only in France there exist at least four different projects of collective music creation under the CC licence which function in the form of the CBPP: HipHopDomain, Featuring album, CCmixter and Take the Bus.

The history of the CBPP then permits to outline two important aspects: firstly, the Unix experience illustrates the fact that collective development of a product implies not only the sharing of distinct ideas, but also the co-ordination among contributions; and secondly that the development and expansion of the CBPP was made possible by the creation of free licences. In the next section (part 2) we will precise what are the specific features that distinguish the CBPP from the commercial model of production, and we will argue that the CBPP can be more efficient for the production of creative content. In section 3 we will analyse the role of the free licences as coordination mechanisms enabling the coherence of the creation processes within the CBPP.

## **2. Commons-based peer production: creative content development within decentralized and distributed system.**

Today music and software creation take place through both commercial (often firm-based) and CBPP models. The literature on the CBPP [for example Lee & Cole, 03; Bonaccorsi & Rossi, 03; Moon & Sproull, 02; Benckler, 02] often focuses on the higher efficiency of this mode of production, where the efficiency is evaluated in terms of product quality (its adequacy to the users' needs, security and variety). Two main features distinguish the CBPP from the commercial production [Amin & Cohendet, 04]: absence of formal constraints and explicit hierarchy; and relative inclusiveness of projects. In this section we will show that these features provide important advantages for the creative content development, which we consider as knowledge-intensive: absence of formal constraints and explicit hierarchy influences the conditions of work of software developers and musicians (2.1.); and relative inclusiveness of the CBPP permits to benefit from the distributed knowledge (2.2.).

### *2.1. Work conditions: absence of formal constraints and of explicit hierarchy*<sup>8</sup>.

According to a permanent school of sociological thought and behavioural economics the participants' efforts within the production process are determined by the terms of exchange and by the relationships between the parties [Mayo, 49; Akerlof, 82; Akerlof, Yellen, 88]. We argue that the absence of formal constraints and explicit hierarchy in the CBPP influences the conditions in which the work is accomplished and as a result the quality of the product.

One can notice that among the factors which determine the quality of the product, the time for the accomplishment of a goal is of special importance. Whereas commercial mode of production supposes a strict system of deadlines, it is not the case for the CBPP. Time constraints and requirement to finish the work in time sometimes result in the fact that the work can be accomplished in a slapdash manner. Talking about the role of job conditions, one of the OSS developers notes:

“...I really like when the code is made well, when it is of high quality, I don't like to do the things in a hurry. I like the code to be planned and possible to maintain. I realize that in the

---

<sup>8</sup> In this section our conclusions are largely based on the study realized in July 2005 by Nicolas Auray and Michael Vicente, ENST, Paris [Auray & Vicente, 2005]. Despite that this study is based on the interviews with the OSS developers, we suppose that the conclusions can be applied for both, music and software creation.

real world in practice you generally have inverse situation. It means that you always have your code at the last minute, you are always pressed, and you don't know the future of your application, how it will be consequently developed, and thus you can't prepare it. Often you are obliged to cut the code." [Auray & Vicente, 05, in author's translation]

In other words, within the CBPP the task accomplishment corresponds rather to the creation of a product of high quality, than to the realization of the work in time. In these conditions even in the case of complex productions, such as software development, more stable and secure products can be created.

Another factor which can influence the quality of the product is the guidance of the production process that is: how the tasks are set. Under explicit hierarchical structure of production, the agenda often comes from the "top" in the form of a command and often these decisions are not negotiable. As a result, in the commercial mode of production, the creator is dispossessed from his creation, and that can negatively influence the quality of the product. In the CBPP the tasks are identified by the participants and the management of projects is ensured by authority<sup>9</sup>. This means the acceptance of the leadership by the participants under two conditions: contributors have the right to choose themselves a task to accomplish as opposed to being planned; and all the leaders' decisions are negotiable and can be turned down if the majority of contributors do not agree<sup>10</sup>. The role of the leader is really significant; it consists in certain goal-setting, in contributions selection and in distribution of official versions. However the leader does not decide the tasks to accomplish nor the deadlines. Contributors choose any task depending on their own needs and interests; they create the product not only for an abstract user, but also for their own use. Finally, such a "user-producer" model which takes place within the CBPP [Von Hippel, 88] permits to create a product which better corresponds to the needs of users.

In this regard, let us cite what one of the developers says about the OSS development:

"It's completely different from the professional development where the features come from marketing, here the features come from where the people do need them... and I think that the great advantage of the Free it is when the features come, in contrast to the commercial applications, they generally have quality and finishing work which has nothing to do...

---

<sup>9</sup> One of the main features distinguishing authority from hierarchy is the negotiability of the decisions; among others features the nature of authority can be also mentioned [Garrouste & Saussier, 05].

<sup>10</sup> Commercial firm-based model of creative content development implies sometimes some elements of authority, but the role of explicit hierarchy remains indispensable.

Because... the culture of professional development it is deadlines unrealistic, the crunch in the end in order to accomplish the task, ...there is also the question that we are not pressed, there is not a person to press, there isn't the project management, there isn't a project manager, there isn't actually hierarchy. There is a person to push me to a certain direction, there is the veto right, and clearly there are senior developers participating during long period who can block the features or the modes of implementation." [Auray & Vicente, 05, in author's translation]

In summary, we can see that activity within CBPP is very different from the commercial model from the point of view of the conditions in which the work is accomplished. In these conditions the content development corresponds with the *creation* rather than *production* process, where the creation is not alien to the creator and where the participants can realize themselves through their work. From this perspective even software development can be presented as a sort of artistic craftsmanship comparable to music creation:

"...They are enamoured of their work, they are enamoured of the work well-made. ...If you want it is somewhat like an artist, it is not already just a technical object, but it is an intellectual creation...You know there are lot of people who think that to write programmes is a job like any other, and I rather agree with this idea, but...the real Geek they think that have a beautiful line of code it's something fantastic... they appreciate it very much, and it is especially thanks to this that finally you have a work of very good quality..."- says an OSS developer. [Auray, Vicente: 05, in author's translation]

## *2.2. Inclusiveness of projects and taking advantage from distributed knowledge.*

In contrast to the commercial model of creative content development, the number and quality of contributions in CBPP are not constrained by the number of employees that a firm can hire or the quality of the given competence pool: "...connecting a large amount of individuals around a common project and without any closure, ... is a way to take advantage of a fantastic potential of distributed skills" [Jullien & Zimmermann, 06]. First of all, the membership in the case of CBPP is relatively open and is based on free cooperation as opposed to the contract-based relationships. In this regard the CBPP is often considered as an effective solution for the knowledge-intensive productions because it better (than a firm) copes with the "identifying and assigning of human capital": CBPP loses less information about who is the best person for a given job might be [Benckler, 02].

Secondly, the Internet serves as a knowledge-creation platform "with which users conveniently and cost-effectively transmit digitized information such as software code, text,

picture/image, voice, and video” [Lee & Cole, 03]. The use of such a platform enables the visibility of the whole creation process by allowing for interaction with a large number of informational resources.

As a result, participants with different skills and experiences, geographically<sup>11</sup> and organizationally dispersed can exchange their *works-in-progress* while gaining “extra sets of eyes to catch mistakes, identify problems, and improve quality” [ib.].

To summarize, the CBPP can be presented as a specific model of innovation, of accumulation and of exchange of knowledge. Despite the lack of exhaustive empirical evidence on the superiority of the product quality developed in the CBPP, the fact that the OSS, as one of the most well-known examples of the CBPP, is used by a large number of enterprises indicates the potential of the CBPP mode [Foray & Zimmermann, 01; Lang, 00].

Two principal characteristics distinguish the collaboration within the CBPP from the commercial production: CBPP supposes authority as the *main* mode of regulation, and high inclusiveness of the projects. On the one hand, in these conditions it is difficult to predict and to influence the development of a product. But, on the other hand, if the collaboration among participants is strong CBPP may increase significantly the quality of a product.

### **3. Free licences as coordination mechanisms: case studies of the Debian and Take the Bus projects<sup>12</sup>.**

We have seen in the previous section that the CBPP corresponds to the decentralized and distributed system of production. It was also argued that these characteristics underlie the creation of high quality products (quality often superior to that of analogous products developed in the commercial firm-based mode), because they provide the conditions for creative work and

---

<sup>11</sup> The fact that participants in the CBPP are usually geographically dispersed doesn't deny that depending on the task in some situations “physical” face-to-face contacts are indispensable. For example, in the Debian OSS project the role of collective meetings is crucial. However, in contrast to the firm-based model these “physical” contacts are of periodic nature.

<sup>12</sup> Our case studies are based on the interviews with the OSS developers [Auray, Vicente: 05], informal discussions with the leader of Take the Bus and analysis of the related literature.

knowledge sharing. However, to achieve effective cooperation, it takes as well complex processes of individual and collective learning.

The enhancement of cooperation is facilitated in a context of common culture. This “shared context” is more than just shared information: it provides simultaneously the method, context, language of learning and determines the evolution of both group and individual competences [Johnson, 92]. Finally, but most importantly, the processes of “enculturation”, that result in the shared practices and routines, contribute to the formation of consensus [Daft & Weick, 84], or in other words, to the coherence of production. This argument is broadly used to explain the formation and efficiency of the Firm as a “learning” organization [for example in Argyris & Schoen, 78; Hodgson, 98; Johnson, 92], because it corresponds to “more durable and integrated characteristics” (than the market) [Hodgson, 98]. For our concern, we raise the question of how this “enculturation” is ensured within decentralized and distributed systems of the CBPP.

The history of evolution of the CBPP provides some evidence that the free licences play an important role in its development and expansion. Moreover, today licences remain almost the only formal coordination mechanism mobilised for this mode of production. We will consider in this section how the GPL and CC licences can contribute to the coherence of product creation in the absence of formal constraints and hierarchy, and in the absence of clear organizational boundaries.

To this effect, we will rely on two different projects that exemplify the CBPP: the Debian project in the sphere of software development (GPL); and the Take the Bus project in the area of music creation (CC). Firstly, we will describe these projects in a nutshell (3.1.). Then, we will show that in both cases of music and software these licences provide conditions for the development of individual and collective learning: in the absence of explicit hierarchy, these licences enable the establishment of efficient authority structure (3.2.); in the absence of clear organizational boundaries, GPL and CC enable the implementation of particular two-tier internal organization based on the Legitimate Peripheral Participation (LPP) principle (3.3.).

Finally we will summarize the common and distinctive aspects of the coordination by the GPL and the CC licence (3.4.).

### *3.1. Debian and Take the Bus: projects' description.*

Despite the idea that in both cases of software and music production, one can talk about the creative content development, and the fact that the Debian and the Take the Bus projects both exemplify the CBPP with its specific features described above (section 2), some important differences lie between them. One of the most important differences concerns the size of the projects and their life period: the Debian project exists since 1993 and involves thousand of participants; the Take the Bus project was initiated only in September 2005 and gathers thirty artists. However both projects are to a certain extent representative in their domains and, hence, permit us to reach some general conclusions about the CBPP functioning.

#### *3.1.1. Debian.*

The Debian project, published under the GPL, was initiated in 1993 by Ian Murdock, a student from Purdue University. This project was the first attempt to create a complete operating system distribution based on the Linux kernel. Debian operation system includes a large number of applications and 1400 different software components, called “packages”<sup>13</sup>. The development of Debian consists in the accomplishment of different tasks such as source code development and packaging, support of the server infrastructure, translation of documents and web pages, development of specific management tools, etc. Also the users providing information on the failures in software in the form of patches and bug fixes play an important role for the project's development. Packaging remains one of the most important activities within the project and most of the developers work on it. Packaging does not suppose necessarily the writing of a source code (sometimes the developers use the code developed by others<sup>14</sup>), but the source code itself is just a sort of a prepared raw material, whereas packaging corresponds to the creation of a software ready-to-use for concrete purposes. The integration of a package to the distribution is a complex operation which imposes many constraints: software should correspond to the GPL licence's

---

<sup>13</sup> Package is the software or a part of the software which is prepared to be implemented to the distribution.

<sup>14</sup> According to [Auray & Vicente, 05] 43% of developers have developed the source code and, then, made its packaging.

terms, and to a large number of technical features to ensure the package's compatibility with other parts of the Debian distribution.

During the period of the project's existence, eight distribution's versions were officially published and the ninth is expected in December 2006. The fact that today Debian unifies almost thousand of developers all over the globe and over the years remains the most popular free distribution of the Linux operating system exhibits the consistency of the development process in Debian.

### 3.1.2. Take the Bus.

The Take the Bus project is relatively new as well as the whole CBPP initiative in music (the oldest project, CC Mixer, exists since 2002). Take the Bus was initiated in 2005 by Philippe Chavaroché and gathers today thirty musicians from different collectivities and countries. The project is published under the CC licence, which imposes the respect of the paternity right, does not permit the use of the product in a commercial way and forbids to modify it (and hence does not contain a share-alike clause).

During the life period of Take the Bus, fifteen music compositions were collectively created and all of them are available on different music platforms on the web. According to the statistics published on the Musique-libre.org website, only on this platform the last composition of Take the Bus settled one month ago has been already downloaded and streamed 550 times. The composition settled one year ago was downloaded and streamed 980 times, which is more than the average number of times on this platform (the average is 840 times). At present the Take the Bus project has already gained recognition within the free music movement and several new compositions are in the pipeline.

The development of music in this project proceeds through several stages. Firstly, the leader of the project provides some information about the tempo and the lyrics that serve as a common base. On the basis of this information contributors create some pieces of music and send their folders to the leader, whose most important role consists in the selection and final arrangement of numerous folders in order to create a single music composition. For their part, contributors create collectively the cover for the final composition and provide it with the tags to

signal the terms of the selected license for the future users. The next step is to upload this “package” on the server and to diffuse it on the different platforms. At this stage the resulting creation becomes a topic discussed among the listeners (members of the project and others). Then, the composition evolves on the basis of the remarks and propositions received.

The main feature of the Take the Bus project which distinguishes it from other music projects based on the CBPP is that the sphere of its application is not limited by a concrete musical style<sup>15</sup>: all the compositions created within Take the Bus belong to different musical styles. We think that this feature is of particular importance, because it indicates that the application of the CBPP in music can be manifold. From this viewpoint we propose to analyse the functioning of the Take the Bus project as a representative example of the CBPP in this area.

### *3. 2. Free licences and the efficiency of authority: the cases of Debian (GPL) and Take the Bus (CC).*

As was noted in section 2, the CBPP does not suppose any formal constraints or explicit hierarchy to ensure the production continuity and coherence. We will show in this section that authority mode of regulation based on the collective recognition of the leadership, on the respect of common rules and on the voluntary joint distribution of roles in the projects is, at least partially, enabled by the GPL and the CC licences. These licences act at three levels: (i) they serve as an explicit expression of common norms to respect; (ii) they permit the “emergent” distribution of the roles according to task performed by a participant; (iii) they direct the actions of participants in the conflict situations. We will show also that whereas the CC licence acts in a similar manner as the GPL for the first two aspects, it represents an alternative approach to the conflict resolution.

#### *3.2.1. Debian.*

Within the Debian project the authority is assumed by its leader, but the role of hundreds of packages’ maintainers is also significant. The role of the leader and of the maintainers consists

---

<sup>15</sup> For instance, CC Mixer specialises only in the electro music, HiphopDomain creates rap sound and Featuring album produces French Music.

in providing evaluations, selection and retention of contributions from other developers and users.

(i) First of all, GPL is the political initiative, so the choice of this licence by initiator signals the main principles of the subsequent functioning of the project. Among these principles one can mention not only a complete disclosure of the software creation process, but also a relatively high level of internal democracy. The latter is represented in the Debian project by the collective election of the leader and by the joint decisions making on the integration of contributions to the distribution.

When the new project is launched, its initiator assumes its leadership, however during the project development another person can be elected. Collective leader elections mean that “anybody who is already admitted to the project is not only eligible to vote in this election but eligible to nominate themselves for the leadership position” – says the current leader of Debian, Branden Robinson [Auray & Vicente, 05]. So, while “hierarchy is founded on the idea that there is an identity between authority and ownership” [Garrouste & Saussier, 05], authority in the Debian project is founded on the collective recognition of the leader’s competences which determines its acceptances by all the members.

Then, the integration of individual contributions to the Debian distribution is based on the system of recommendations from many other developers (sponsorship) and not on the leader’s individual decision. So, contributors are introduced not only to the development of specific tasks they are working on, but also to the evolution of the whole product.

From this perspective the GPL provides the basis for the cooperation development, because all the members already admitted and potential have a clear idea about the norms to respect by both the leader and other participants. GPL acts as a signal that clears potential ambiguities about individual behaviour and respect of the social norms; and if the intention is made explicit, deviation may be seen as an obvious and explicit challenge to social norms and, so, may be sanctioned by the project’s members [Arora and al., 04].

(ii) Secondly, the paternity principle expressed in the GPL supposes that all the contributions to the Debian creation are not anonymous: for example, each package must contain the name and the email of its author. In the framework of the project, developers identify

themselves and each other according to the package they have made. But to make a package is not just to create software; it includes also its updating and support along with the evolution of the main of the project (because as was noted earlier the software components of the operating system are technically interdependent). The GPL stimulates a specific process of the distribution of roles in the project: as opposed to being assigned to carry out a task in terms of commercial production, the roles in Debian emerge depending on the task performed by a participant. In contrast to the commercial model of software development where the activities are usually separated between different actors, the author realizes the work from A to Z, because he is the main responsible for the task performance in the project. That is why 78% of the Debian developers maintain their own packages [Auray & Vicente, 05].

Furthermore, at the level of interaction, maintainers are encouraged to cooperate intensively with other developers and users and to take into account their remarks and propositions, because these critics permit maintainers to learn how to improve their submissions. This is not always the case in the commercial software firms where the tasks are separated and where it may be difficult to identify at which stage of the software development the error appeared: “there are great incentives...for individual employees to conceal rather than reveal error or weakness in their work” [Lee & Cole, 03].

(iii) Finally, the GPL licence allows to run the program, to study how it works, to modify and to redistribute it, and in such a way illustrates a specific approach to the problem of conflict resolution. In the situations when the consensus cannot be found among developers, they have “the exit option” – the right to leave the project and to initiate their own project in parallel (the fork) [Egyedi & van Wendel de Joode, 04]. On the one hand, the initiation of a fork can negatively influence the continuity of a project’s development. But on the other hand, this option permits in some cases to avoid the excessive use of the “voice option” (the direct expression of dissatisfaction) [Hirshman, 70]. In the extreme cases when during an important period of time the negotiation processes do not manage to solve the conflict, the option to exit permits to reduce the number of conflicts emerging in Debian and go on with the work in progress (moreover due to the copyleft principle, the fork will remain open).

### 3.2.2. Take the Bus.

In the Take the Bus project, authority is represented by the project's leader, who is at the same time the initiator of the project.

(i) The process of the choice of the licence is different than in the case of the Debian project, where it corresponds to the individual decision of the initiator of the project. In contrast to Debian, in the Take the Bus project the licence, or more precisely the *terms* of the licence, becomes itself the result of a negotiating process between the leader and potential contributors. To initiate the project, the leader provides some information about the tempo and the lyrics that serve as a common base. On the basis of this information potential contributors make their choice whether they want to participate or not, and express their opinion about the licence terms. Following the discussions with the potential contributors the project leader selects the licence for the final composition. Once the consensus among the potential participants and the leader is reached, the production starts.

These negotiations permit to prevent some tensions (as, for example, the problem of how the individual contributions will be subsequently used) when the creation process has already started up. Moreover, collective choice of the licence's terms, in particular the establishment of the non-modification clause, signals the acceptance and the recognition of the leader by contributors.

Thus, despite the fact that the process of the licence choice differs from this process in Debian, the role of the CC licence is *in fine* very similar: just as the GPL, CC acts as a basis for the cooperation development; as an explicit expression of some basic common norms in the project, accepted and respected by both the leader and the contributors.

(ii) In Take the Bus the CC licence contains the attribution clause (the principle of paternity). As a result, like the GPL, it stimulates the emergence of the roles in the project according to the task performance and eventually encourages the cooperation among project's participants and users in a similar manner as does the GPL. The fact that all the contributions are not anonymous means that all participants of the Take the Bus project, its leader as well as its contributors, are collectively responsible for the final music compositions which will be largely diffused on different web platforms. Thus the leader is interested to cooperate with other

participants, direct contributors and users, to participate in the afterwards discussions on the forums, and to take into account the remarks and propositions. At the same time as the names of all contributors are indicated in the project, they are encouraged to support their collective project while providing it with the evolutions like video or texts. In such a way, similarly as the GPL stimulates interaction among participants, active and not, the CC licence encourages the contributors not only to propose their ideas, but to contribute to the project's maintenance and support. So, while the initial proposition of a contribution can be based on a distinct individual activity, project's maintenance and support call for the active cooperation with others.

(iii) In the case of conflicts within the project, the CC licence acts inversely as the GPL: a contributor or a group of contributors who do not agree with the decisions of the leader and other contributors do not have the right to initiate in parallel their own project on the basis of music compositions developed in the framework of Take the Bus. The absence of this right is fixed by the non-modification clause on the CC licence (which was collectively chosen by all the project's participants). So, in contrast with the Debian project, CC in Take the Bus permits to keep the project's integrity and continuity in its initial form even in situations when the consensus cannot be rapidly reached.

### *3.3. Introduction of newcomers and LPP principle: the cases of Debian (GPL) and Take the Bus (CC).*

Connecting a large number of participants around the common project is one of the key factors determining the efficiency of the CBPP, however high inclusiveness of the projects in the CBPP could result in a lack of production coherence, because “the complexity and communication cost of a project increase with the square of the number of developers, but the amount of work done only rises linearly” [Brooks, 95]. We will show in this section that the free licences provide the basis for the specific two-tier organizational structure or the “Legitimate Peripheral Participation” (LPP) [Lave & Wenger, 91]: an organization form based on a limited core and a large open periphery. The role of the GPL and the CC licence is twofold: (i) on the one hand, they permit learning by contributing of a large number of participants; (ii) on the other hand, they permit to separate the core and the periphery of the projects and to cope with the

complexity of task coordination. However, while separating between the core and the periphery CC creates no additional barrier to overcome to become the core contributor, and participants are freer (than in the case of Debian) to choose between the participation in the core or at the periphery

### 3.3.1. Debian.

The core of the Debian project is represented by the official developers (as the leader and maintainers) whose contributions have already been accepted for the Debian distribution. However thousands of non-admitted participants contribute to the project in the periphery realizing some simpler tasks such as patch making, debugging or translation. These non-admitted participants do not have the right to participate in the leader's elections nor to apply themselves to the leadership position. In spite of this duality of treatment, the core relies on the periphery in order to select and retain among different propositions to produce an official release. The logic of the project functioning is based on the idea that, initially, developers may join the project and learn at the periphery, and as they become more competent they move to the core of the project. For example, the integration of a new contribution to the distribution cannot be validated by an individual decision of a leader or any of the maintainers. In order to become an official member, the newcomer has to establish the contacts with active developers through his participation at the periphery. When these contacts are made, the newcomer can ask an active developer to sponsor him. For some time, the sponsors check the technical features of the package proposed by the newcomer. Furthermore, the sponsors submit the newcomer to specific tests to make sure both of his ability to enter the project, and if his ideas correspond to the ideology of Debian. Finally, the sponsors express their opinion concerning the possibility of integration of the newcomer's contribution to the Debian distribution and, consequently, his admittance as an official member [Auray & Vicente, 05].

(i) The continuation of the code visibility guaranteed by the copyleft principle of the GPL permits the participants to "learn from their own and others' prior successes and failures, they can sort themselves into tasks appropriate to their skills, move up to more challenging tasks, and/or generate better variations of the source code" [Lee & Cole, 03]. The GPL allows to study how the

program works and, newcomers have the possibility to learn not *from* talk, but to learn *to* talk as a key to LPP, and this orientation has important advantages of learning in context [Lave & Wenger, 91]. “Legitimate peripheral participation provides a way to speak about the relations between newcomers and old-timers... A person’s intentions to learn are engaged and the meaning of learning is configured through the process of becoming a full participant in a socio-cultural practice. This social process includes, indeed it subsumes, the learning of knowledgeable skills.” [ib.]. In other words the GPL enables the gradual process of learning through contributing, which is not limited by the boundaries of an enterprise as in the case of the commercial software production. It is worth noting that in contrast to commercial firm software production, the two-tier structure guarantees the possibility of a dramatic reduction of the time necessary for the development of cooperation among active participants: the cooperation between new-comers and old-timers takes place as soon as the former contribute at the periphery and not when they come to the core.

(ii) Furthermore, the visibility of the project’s functioning and evolution also imposes some duties to the newcomers. A newcomer has to study thoroughly the project *before* making his propositions; otherwise he gets immediately the message *rtfm*: “read the f... manual”. The GPL not only stands guard over the possibility to gradually learn through the contributing, but also plays the role of a barrier to entry to the core. But, in contrast to commercial model where the cooperation takes place only when a contract is made, this barrier is not identified by “abstract” competencies of newcomer, but by his competencies in accordance to the concrete project with its norms and routines. Hence, at the level of integration to the project of newcomers, GPL permits to find equilibrium between the afflux of new contributions and the crowding that can negatively influence the integrity of the project.

### 3.3.2. Take the Bus.

On the one hand, the structure of the Take the Bus project is similar to the Debian’s one: the core consists of the leader and the limited number of artists whose contributions were selected by the leader for the official distribution, whereas anyone may contribute to the project’s development at the periphery (for example, through the participation in the forums of

discussions). As in the case of Debian the core relies on the periphery in order to select and retain among different propositions to improve the product. On the other hand, there is a big difference in the logic of functioning between these two projects. For example, in Take the Bus the leader publicly announces the information about the tempo and the lyrics which serve as the common base for potential contributors, thereby newcomers may choose between starting to learn at the periphery or sending their contributions directly to the leader and becoming at once official members if their contributions are selected.

(i) First of all, the CC enables a publicly available process of collective music creation (as well as its extensions in the form of public discussions or video and texts), because participants can keep their artists status (paternity principle in the licence), and because the contributors have the possibility to control the undesired use of their product (the clause of non-commercial use in the licence). So, as well as the GPL the CC licence provides the opportunities for newcomers to learn from theirs and others' successes and failures, to learn while contributing to the project, to learn in context, and not only through the observation of the results. These learning mechanisms, hence, are not bounded by a particular enterprise, and the process of creation gains the attention of relatively large amount of artists and simple users which make their contributions for the products' improvement.

(ii) The CC licence permits to separate between the core of the project and the periphery through the non-modification clause: the right to select the contributions and to implement any modifications is assigned, under the licence's terms, *only* to the project's leader (even official members do not have this right). This clause solves the problem of task coordination in the Take the Bus project. There is no barrier to overcome between the core and the periphery, and participants can choose between the participation in the core or at the periphery depending on their skills, interests and intentions. While the LPP in Take the Bus supposes less control over the core of the project, higher inclusiveness to the project is to a certain extent counterpoised by the limitation of the participants' rights to make alterations in an independent way.

### *3.4. GPL and CC as coordination mechanisms: summary.*

We have argued that the role of the GPL and CC licences is much larger than just to ensure the sharing of distinct ideas: these licences contribute *also* to the co-ordination of distributed pieces of knowledge, hence to the creation of “commonly shared bodies of knowledge” which are more than just the sum of the individual parts.

In the absence of explicit hierarchy, GPL and CC stimulate the development of cooperation among contributors by creating the conditions in which the authority is effective and accepted by the participants. Initially, these licences represent the explicit expression of the common norms to be respected, as for example the leadership and democracy, which provide the basis for the cooperation development within the projects. Also, while being based on the paternity principle, GPL and CC stimulate the distribution of the roles in the projects depending on the tasks performed by these participants (as opposed to being assigned to carry out a task). Such distribution of the roles makes each contributor responsible for his own task and for the resulting quality of the collective product. That responsibility entails cooperation among participants in order to accomplish the common goal, create collectively a product of high quality.

In the absence of clear organizational boundaries and in the presence of a large number of participants, the GPL and CC licences permit the implementation of a specific two-tier organizational structure. Due to this structure the core of the project can be separated from the periphery where any participation is eligible. These licences make it possible to benefit from the participation of a large number of participants and to cope with the coordination complexities<sup>16</sup>.

However, despite the fact that for some tasks GPL and CC act similarly, for others they represent alternative approaches. Among such tasks, we can distinguish the following two:

- GPL and CC co-ordinate differently the actions of projects' participants in the conflict situations: CC makes the accent on the insurance of the continuity of authority in its initial form, whereas GPL permits the initiation of the project in parallel.

---

<sup>16</sup> Even in the presence of the selection principle in the CBPP, this approach differs from the commercial contract-based, because all comers can participate at the periphery, hence, the information about “human capital” is not lost.

- GPL and CC represent also different approaches to the control over the core of the project: GPL creates a sort of a barrier to the entry of new participants to the core of the project, whereas CC encourages the contributions from all comers in the core and at the periphery.

To interpret this difference we resort to the idea developed in the literature on production coherence, that is the way coherence is achieved depends on the characteristics of production process. So, in the following part we will differentiate between the processes of music and of operating system development. We will discuss whether the differences in the GPL and CC coordination meet the specificities of the production processes in both cases, and, thus, whether these licences can be really considered as coordination mechanisms stimulating production coherence.

#### **4. Free licences and the content specificity.**

Music and software contents are very different in their nature: for example, software represents an example of the functional work, which can enable the subsequent creation of value; and music is usually in itself the “final product”, which is aimed at the satisfaction of the users’ needs. So, one can suppose that there may be also important differences in the software and music production processes. Following the evolutionary literature on the problem of coherence of production process [Dosi and al., 92] we propose to differentiate between the music and the software productions according to the following three characteristics:

- complementarity of assets;
- convergence of production processes;
- learning.

First of all, we will describe these three characteristics in the view of collective music and software development (4.1.), and then we will discuss whether the GPL and CC licences meet these specificities of the characteristics of the production processes (4.2.).

#### *4.1. Description of characteristics of the production process.*

##### **Complementarity of assets.**

High level of complementarity means that the value of already invested assets increases with the appearance of new products or procedures. We consider this characteristic to apply to the music and the software commons-based creation, because both of them correspond to the knowledge-intensive production and imply the sequential and complementary innovation processes [Bessen & Maskin, 00]. In other words, in both cases of collective music and software development, the value of contributions already realized increases with new contributions.

##### **Convergence of production processes.**

The convergence property, as another aspect of path-dependencies, intends that there may be confluence between two or more productions and that this reciprocal influence will to a certain extent determine the organization of the concerned productions. For the case of commons-peer production, this characteristic can be interpreted as the interdependence of contributions realized within the project. We can say that contributions in music creation are more autonomous than in operating system. In the case of music creation, while working on the distinct parts of future musical composition, musicians do not specially care about what other members do. The final arrangement, realized by the leader, consists in the organization of these parts in a particular sequence and in their processing in order to ensure the compatibility between coterminous fragments in this sequence. In the case of the operating system, developers have to coordinate their efforts with a large number of other developers at different levels of production process: for instance, in order to add new application or a package to the operating system it has to be compatible with other applications at different levels.

##### **Learning.**

Individual and collective learning is one of the most important aspects of creation activity, because it results in the dynamic development and evolution of organisational norms and routines. We can also find differences between the music and software development in terms of the place of individual and collective learning within the process of content creation. For the music creation, interaction and collective learning among members play the most important role in the stages when the project is relatively advanced (forums, remarks and evolutions), whereas

initial contributions often imply individual learning processes. In the case of the software creation, individual and collective activities are strongly interdependent in all the stages of product creation: to do a valuable contribution it is necessary to study thoroughly the project, and this already implies an interaction with the other developers (new developers often have to start with some simpler tasks in order to learn gradually the project's functioning and only then to propose their contributions.) That's why for example, the periodic "physical" meetings among the Debian developers play a very important role for the creation process, and it is not the case for the Take the Bus participants.

#### *4.2. GPL and CC and the characteristics of the production processes.*

Taking into account the three characteristics of the process of music and software contents, we propose the interpretation of the difference in the respective licences' structure:

- Due to the high complementarity of contributions in both cases, GPL and CC enable specific authority structuring, based on the negotiation processes among participants and on the distribution of the roles in the projects depending on the proposed tasks; as well as they both enable the establishment of the two-tier organizational structure. These two aspects result in the mobilisation of a large number of contributions for the collective product development.
- The additional control for the access of newcomers to the core of the project is of no need in music creation because of the relatively low level of interdependence of contributions in the initial stage of the creation of music compositions. In contrast to software, the potential problem of crowding (which can take place in the case of the operating system creation) is almost eliminated in music projects. That's why the CC licence does not play the role of a barrier to entry to the core as does the GPL.
- Finally, higher level of interactions of developers (and hence collective learning) results in the development of a larger number of tacit and codified routines and procedures which underlie the creation of the Debian operating system. These conditions lead to the situation where the imitation of initial project (by forking) is complex. Hence the continuity of the project in its initial form is protected by the process itself. In contrast, in music creation the imitation and

splitting up of the project is less complex, and in order to achieve higher coherence of the project it is important to ensure an additional control over project' continuity in its initial form.

In this Table, we summarize how the structure of the licences corresponds to the characteristics of the production processes:

	Software development and the GPL	Music development and the CC licence
<b>CONVERGENCE</b> <i>Barriers for introduction to the core</i>	+	-
<b>COMPLEMENTARITY</b> <i>LPP + negotiated leadership and "emergent" roles</i>	+	+
<b>COLLECTIVE LEARNING</b> <i>Control over the project' continuity</i>	+	-

### Conclusion.

It was argued in the paper that the CBPP can be very effective for the development of creative content. However, this efficiency calls for some specific coordination; otherwise the development process faces the problems of production discontinuities and lack of coherence. We set up the hypothesis that the role of the free licences can be much larger than it is usually presented in the literature: free licences not only allow knowledge sharing, but they can also serve as coordination mechanisms stimulating individual and collective learning within the development process. We have used the knowledge-based approach to verify our hypothesis and

to show that: first, in both cases of music and software creation, the free licences stimulate individual and collective learning while providing the basis for the “shared context” necessary for the evolution of both group and individual competencies; and, second, that the free licences meet the specificities of the production processes.

We have analysed two examples when the CBPP is applied for the software and music collective creation: The Debian and the Take the Bus projects which can be considered as representative in their respective domains. The comparative study of these two projects permits us to do some general conclusions about the CBPP functioning. First of all, we can say that the free licences determine the process of the structuring of leadership and of repartition of the roles in the projects, while allowing the efficiency of authority as the main mode of regulation. Secondly, these licences permit the establishment of the two-tier organizational structure due to which the CBPP benefits the participation of a large number of participants and copes with the coordination complexities. Finally, but very importantly, the structure of free licences is adopted to the particular characteristics of the production processes such as complementarity of contributions and their convergence, and the repartition between the individual and collective learning.

Our analysis must be considered as a first step and far more elaborated study is necessary for the consideration of the role of free licences within the production process. One of the possible directions for future studies is the implementation to the analysis the aspects of actors incentives and to consider free licences as a particular regime of IPR. Our starting point for this attempt is the perception that economic analysis has so far refrained from establishing a link between the evolution of cooperation and IPRs as two important aspects of knowledge creation. Some studies on the CBPP [for example Maurer & Scotchmer, 06] emphasize that free licences stimulate incentives in the strategic behaviours of actors that are distinct from earlier uses of intellectual property and remedy some defects of property protection (for example, the encouragement of ideas’ disclosure). Further, it would be interesting to analyse the influence of this specific incentive structure to the dynamic processes of individual and collective learning. Taking into account the link between the evolution of cooperation and IPRs will permit to broaden our problematic and to consider whether the IPRs can serve not only as a mechanism

guiding the allocation of resources (as “pure” *incentive mechanism*), but can also contribute, as a *learning mechanism*, to the *process* of creation of common resources. To analyse this question can be important for the study of essential economic issues such as IPR’s efficiency evaluations.

## References.

- Akerlof, G.A., 1982. Labor Contracts as Partial Gift Exchange. *Quarterly Journal of Economics*, 97, 543-569.
- Akerlof, G.A., Yellen J.L., 1988. Fairness and Unemployment. *American Economic Review, Papers and Proceedings* 83(2), 44-49.
- Amin, A., Cohendet, P., 2004. The architecture of Knowledge: communities, competences and firms. Oxford University Press. Oxford.
- Argyris, Ch., Schön, D., 1978. Organizational Learning: A Theory of Action Perspective. Reading, MA: Addison-Wesley.
- Arora, A., Fosfuri, A., Gambardella, A., 2004. Markets for Technology and their Implications for Corporate Strategy. in: Reichman, J., Maskus, K. (Eds.), International Public Goods and Transfer of Technology Under a Globalized Intellectual Property Regime, Cambridge University Press. Cambridge.
- Auray, N., Vicente, M., 2005. Empirical study on the Open Source Software realized in July 2005 by Nicolas Auray and Michael Vicente. ENST, Paris.
- Benkler, Y., 2002. Coase's Penguin, or, Linux and The Nature of the Firm. *Yale Law Journal*, 112, pp. 369-446.
- Besse, J., Maskin, K., 2000. Sequential Innovation, Patents and Imitation. MIT, DEPT of Economics, Working Paper N°00-01, January.
- Bonaccorsi A., Rossi C., 2003. Why Open Source can succeed. *Research Policy*, 32(7), pp.1243-1258.
- Bonaccorsi, A., Rossi, Ch., 2003. Why Open Source software can succeed. *Research Policy*, N°32, Issue 7, pp. 1243-1258.
- Brooks, F.P., 1995. The Mythical Man-Month: Essays on Software Engineering. Reading. M.A.: Addison-Wesley.
- Cohendet, P., Llerena, P., 2001. Routines and the Theory of the Firm: the Role of Communities. DRUID Conference, Aalborg, June.
- Daft, R., Weick, K., 1984. Toward a model of organizations as interpretation systems. *Academy of Management Review* 9(2), pp. 284-295.
- Dosi, G., Teece, D. J., Winter, S.G., 1992. Towards the theory of corporate coherence: Preliminary remarks. in: Dosi, G., Gianetti, R., Toninelli, P.A. (Eds), *Technology and Enterprise in a Historical Perspective*. Oxford University Press. Oxford.
- Foray, D., Zimmermann, J.-B., 2001. L'économie du logiciel libre: organisation cooperative et incitation à l'innovation. *Revue Economique*, Vol. 51, pp. 77-93
- Garrouste, P., Saussier, S., 2005. Looking for a theory of the firm: Future challenges. *Journal of Economic Behavior & Organization*, Vol. 58, pp. 178-199.
- Hirschman, A. O., 1970. Exit, Voice, and Loyalty: responses to decline in firms, organizations, and states. Cambridge, Massachusetts and London: Harvard University Press.
- Hodgson, G.M., 1998. Competence and contract in the theory of the firm. *Journal of Economic Behavior & Organization*, Vol. 35, pp.179-201.

- Johnson, B., 1992. Institutional learning. in: Lundvall, B.-A. (Ed.), *National Systems of Innovation: Towards a Theory of Innovation and Interactive Learning*. Pinter, London, pp. 23-44.
- Jullien, N., Zimmermann, J.-B., 2006. *New approaches to Intellectual Property: from Open Software to Knowledge-based Industrial Activities*. DIME Working Papers on Intellectual Property Rights, N°5.
- Kogut, B., Zander, U., 1992. Knowledge of the Firm, Combinative Capabilities, and the Replication of Technology. *Organizational Science*, Vol. 3, N°3, pp.383-397.
- Lang, B., 2000. *Logiciels Libres et Entreprises*. Terminal, N° 80/81, « Les logiciels libres : de l'utopie au marché », Éd. L'Harmattan. [www.terminal.sgdg.org](http://www.terminal.sgdg.org). (07.10.2006)
- Lave, J., Wenger, E., 1991. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press. Cambridge.
- Lee, G.K., Cole, R.E., 2003. From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development. *Organization Science*, Vol. 14, pp. 633-649.
- Maurer, S., Scotchmer, S., 2006. *Open Source Software: The New Intellectual Property Paradigm*. Hendershott, ed., *Handbook of Economics and Information Systems*, Amsterdam: Elsevier.
- Mayo, E., 1949. *The Social Problems of an Industrial Civilization*. Routledge and Kegan Paul, London.
- Moon, J.Y., Sproull, L., 2002. *Essence of Distributed Work: The Case of the Linux Kernel*. in: Hinds, P., Kiesler, S. (Eds.). *Distributed Work*, Cambridge, MA: MIT Press, pp. 381-404.
- Osterloh, M., Frey, S.B., 2000. Motivation, Knowledge Transfer, and Organizational Forms. *Organizational Science*, Vol.11, N°5, pp.538-550.
- Van Wendel de Joode, R., 2004. Explaining the organization of open source communities with the CPR framework. The 10<sup>th</sup> Conference of the International Association for the Study of Common Property: "The Commons in an Age of Global Transition: Challenges, Risks and Opportunities", Oaxaca, Mexico, August 9-13.
- Von Hippel, E., 1988. *The Sources of Innovation*. Oxford University Press, New York.