INTRODUCTION

The model that will be presented in this paper tries to bring the impure public goods model in a strategic context. On one hand the open source software development is identified in terms of contribution to impure public goods, so that we can handle a low level of free riding and define utility in terms of availability of two goods, one public and on private, strictly related each other.

On the other hand open source software development is taken as a problem of game-theory, where individual's choice depends not only by is own preferences but also by other developer's behavior, letting us thinking in a strategic way.

Starting from this point we can try to define what are general condition and feature that apply to every subject and that underlie their behaviors. In other words the goal is to arrive at a new formulation that take into account both the specificity of development process and the strategic behavior of the actor involved.

In addition, the model takes into account a particular kind of subject: the programmers that are employed by a commercial software house.

Reasons are double: on one hand programmer constitute an important share of the whole population of developer; on the other hand is important to demonstrate the possible cohabitation of two realities: open source software and closed source software.

THE MODEL

In this model we consider only a particular category of individuals, the programmers. Main hypothesis are two:

1. Programmers are paid by closed software firms, so they can contribute to open source software projects only when they aren't at work.

2. Open source software development and closed source software development are considered akin, so, if a subject decide to spent an additional hour in developing

open source software, for our purpose looks like as if he renounces to one hour of extra-pay.

1.1 The case of the isolated agent

We consider the sequent constraint:

$T^i = t_1^i + t_2^i$ where

- T^i is the total time (in hour) spent in software development (either closed and open). We consider T^i steady for every subject.
- t_1^i is the time spent in developing closed source software.
- t_2^i is the time spent in developing open source software.

In other words, t_1^i e t_2^i can vary, but their sum has to be steady. After that we consider X_1^i, X_2^i and X_3^i, that represent the 3 feature associated to t_1^i and t_2^i. In particular:

- $X_1^i = \gamma t_1^i$ where X_1^i is the private feature obtained by the programmer working in the firms. In this way one unit of t_1^i generates γ units of X_1^i; X_1^i represent the private feature that a programmer gain from the development of commercial software. So, in monetary terms, X_1^i is precisely the wage.

- $X_2^i = \alpha t_2^i$ is the private feature that a subject gains from developing open source software. Like before, one unit of t_2^i generates α units X_2^i. In this way X_2^i is the utility generated by the development of open source software, and this utility is considered in terms of utility of use. So X_1^i and X_2^i are not immediately comparable, but we can overtake this problem: software, and in particular software for advanced users, is a kind of good relatively simple from an economic point of view: a program value depend essentially on performance and feature implemented. A kernel market value, for example, is not conditioned by design, imagine, etc. but is based on hard features like cpu, memory and process management. In practice, also due to the high level of expertise of clients, we can assume that the market value is a good proxy of the value of use of the good (we are not talking about a Valentino's dress for which price is justified by other factors!). In the end, if we want to give a monetary value to X_2^i we can say:

  $\$ X_2^i = \$_{P.e.}$ + **Goodwill** where

  $\$ X_2^i$ = monetary value of X_2^i

  $\$_{P.e.}$ = market value of an equivalent software

  Goodwill = surplus justified by the high level of personalization (tailored-software).

- $X_3^i = \beta t_2^i$ is the public feature associated to t_2^i. As a consequence, if we consider X_3 as the level of the whole public feature given by programmers, we will have

  $X_3 = \sum_{i=1}^{n} X_3^i$ and going further $\tilde{X}_3^i = \sum_{i=1}^{n} X_3^i - X_3^i$.

Talking about X_3^i some considerations are needed; if X_1^i and X_2^i are in some way comparable, define a unit of measure for X_3^i is much more problematic. How can we give a univocal value to a public good? Emerge the classical problem of marginal rate of substitution between private and public goods and other problems connected to the net-externality. So we don't assume any kind of unit of measure for X_3^i: we consider it in term of absolute value, simply like a sum of the individual contribution.

As a consequence, one unit of t_2^i generates together α units of X_2^i and β units of X_3^i. So, programmer problem is a classic problem of maximization:

$$\max_{t_1^i, t_2^i} \left\{ U(X_1^i, X_2^i, X_3) \middle| T^i = t_1^i + t_2^i, X_1^i = \gamma t_1^i, X_2^i = \alpha t_2^i, X_3 = \tilde{X}_3^i + X_3^i \right\}$$

Considering community of relevant size, individual contribution will become ever and ever irrelevant if compared to the total contribution, and so we assume $X_3 \cong \tilde{X}_3^i$.
So, the problem of utility maximization can be rewritten as:

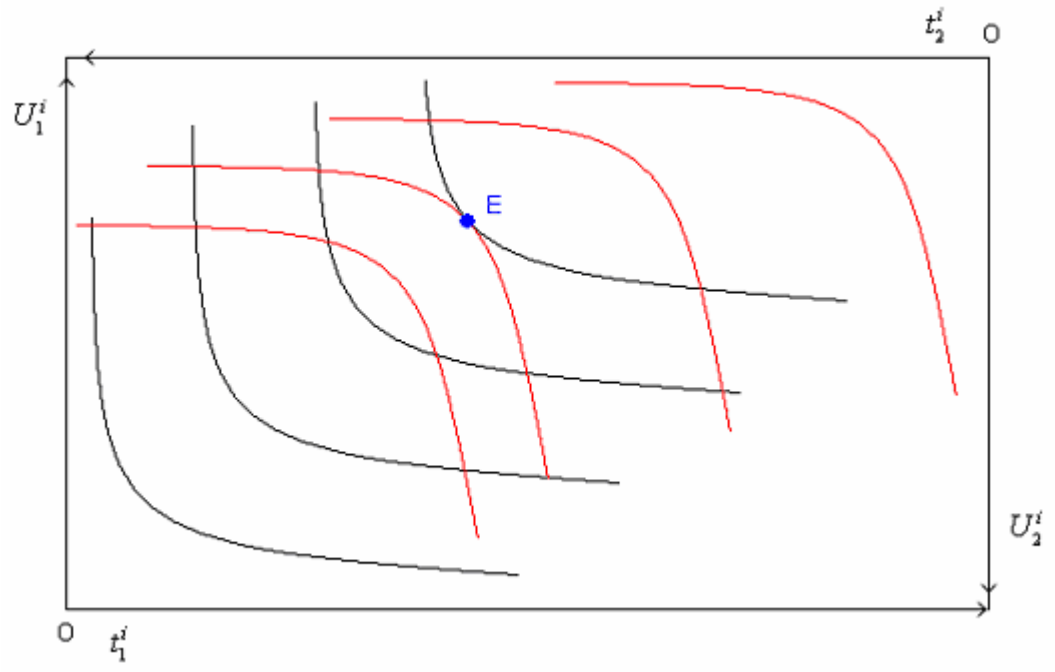
$$\max_{t_1^i, t_2^i} \left\{ U(t_1^i, t_2^i; \tilde{X}_3^i, \alpha, \beta, \gamma \right\}$$

For the sake of simplicity than we take α, β e γ as invariant and $\tilde{X}_3^i$ as given for every individual; as a consequence, the only variables are t_1^i and t_2^i, and that means that the problem of utility maximization is:

$$\max_{t_1^i, t_2^i} \left\{ U(t_1^i, t_2^i) \right\}$$

Going further, if we call U_1^i the utility associated to X_1^i and U_2^i the utility associated to X_2^i, optimum configuration will be a point E in the Edgeworth box such as $U_1^i = U_2^i$.

Fig. 1 Edgeworth box



1.2 Strategic interaction

In this case decision of software development depends not only on individual utility, but also on other agent's behavior. Now $\tilde{X}_3^i$ is taken into account more seriously.

If we call:

- t_s^i time needed for the entire development of an open source project for a subject *i.* You have to notice that this value vary from person to person due to the differences in programming skills.

- π_i, likelihood estimate by the subject *i*, that another agent develop that project.

So, we can distinguish two cases:

a) $\pi_i X_2^i < t_s^i \alpha - t_s^i \gamma$ In this case, is not likely to get advantage of someone work's, because π_i is too low. In this case agent *i* should decide to develop himself. So $t_2^i > 0$

b) $\pi_i X_2^i > t_s^i \alpha - t_s^i \gamma$ Now π_i is sufficient high and it's very likely to get advantage of someone work's. In this case agent *i* is not stimulated to develop and $t_2^i = 0$.

In sum, we can say that for every open source project is necessary considering the two conditions just define, and so:

$$T_2 = \sum_{i=1}^{n} t_s^i \Big| \quad \text{a) is satisfied}$$

Where T_2 is the total amount of hour dedicated to open source projects by the community. As a consequence also

$$X_3 = \sum_{i=1}^{n} X_3^i \qquad \text{with} \qquad X_3 = \sum_{i=1}^{n} \beta t_s^i \Big| \quad \text{a) is satisfied}$$

Where X_3 is the total amount of public feature.

2 Conclusion

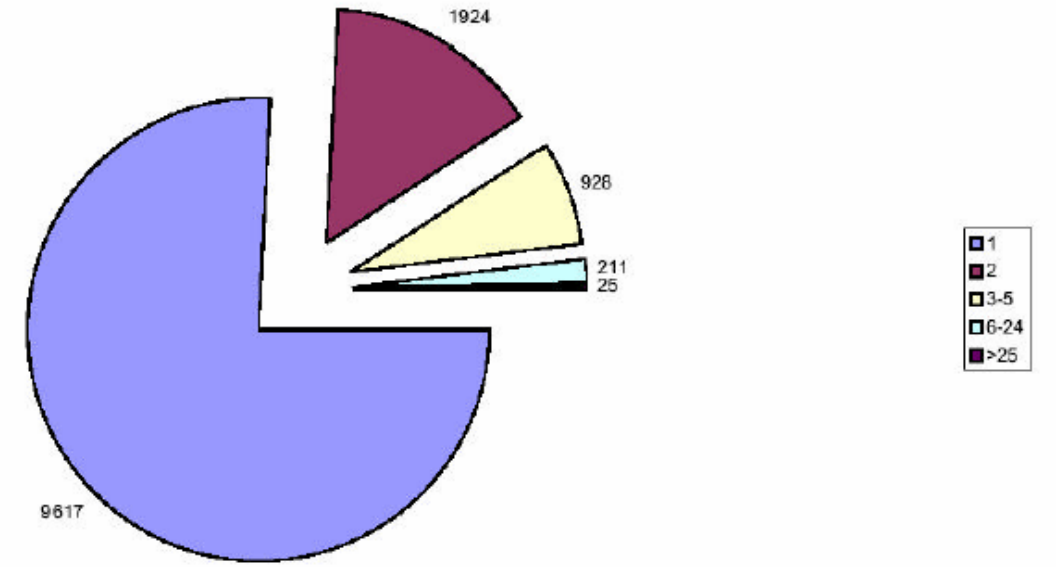
This model considers only a particular category of individuals, the programmers. In this way we consider an important part of the open source community in order to underline how does the development process work. Development is heavy influenced by expert programmers, for whom, utility of personalized application is higher and t_s^i is lower, in contrast to less skilled programmer, more disposed to use standardized application.

Indeed, if we look to the apache project, we can see the importance of most skilled programmers that have contributed for the 83% of the total code.

In the end, the effort is to put theory of impure public goods in a strategic context.

The idea is to consider interrelation between individuals, because every single subject is influenced by behavior and expectations of other people in the community.

Figure 1: Distribution of Contributions by Participant



Source: Lerner J. e Tirole J. (2000), *The simple economics of open source*, NBER Working paper Series, 7600, NBER, Cambridge (Mass).

Bibliography

Andreoni J. (1988), Privately Provided Public Goods in a Large Economy: The Limits of Altruism, *Journal of Public Economics* vol.35, pp.57-73.

Cornes R. e Sandler T. (1984), The Theory of Public Goods : Non-Nash Behavior, *Journal of Public Economics*, vol.23, pp.367-79.

Cornes, R. e Sandler T. (1986), *The theory of externalities, public goods and club goods*, Cambridge University Press, New York

Hann I., Roberts J., Slaughter S. e Fielding R., (2002), Delayed returns to open source participation: An empirical analysis of the apache HTTP server project, www.idei.asso.fr/Commun/Conferences/Internet/OSS2002/Papiers/Hann.PDF

Lakhani K. (1999), *Sustaining the virtual commons: end user support for Apache web server software on the Usenet.* Unpublished Masters thesis, Massachusetts Institute of Technology, Cambridge, MA.

Lerner J. e Tirole J. (2000), *The simple economics of open source*, NBER Working paper Series, 7600, NBER, Cambridge (Mass).