# CommunesPlone, an original open-source model of resource pooling in the public sector.

Robert Viseur, Nicolas Jullien

# CommunesPlone, an original open-source model of resource pooling in the public sector.

Robert Viseur, Université de Mons, Robert.Viseur@umons.ac.be
Nicolas Jullien IMT Atlantique, LEGO Lab. & Università Ca'Foscari, Nicolas.Jullien@imt-atlantique.fr

**Abstract**—This article studies the building of a successful open-source project for the public sector, named CommunesPlones. It proposes an original institutional arrangement to onboard different users, especially those unable to contribute in code, for them to express their needs and provide the resources needed to develop and maintain the software addressing them. It has created an open-source editor collectively owned by its users. This pooling by a trusted third party model may appear as an alternative to the classic FLOSS projects, when users have few development skills, even if the consequence is that, although the code is still open, the development is centralized by this special type of open-source editor.

**Index Terms**—FLOSS, pooling of resources, public sector software, governance

## 1 Introduction

Modern software solutions, for obvious cost reasons, are shared by several users. This may be achieved by supply pooling (a publisher creates a solution and sells it to "clients") or by demand pooling (a group of actors coordinates to develop a resource that addresses shared or complementary needs), with several intermediate solutions (e.g. a group of actors mandates a provider to develop a solution for them).

The possible technical, organizational, and economic arrangements vary according to who allocates and who pays for the human and technical resources needed to develop two main capacities [1]: the capacity to specify users' needs into technological requirements and code, and the mastery of the development process (prioritizing and integrating the code into a coherent solution).

This article studies one such arrangement, the FLOSS project CommunesPlone, which has developed several dedicated public software solutions based on one of the oldest FLOSS Content Management Systems Plone. Initiated by municipalities in the Walloon Region of Belgium, it is also used, however marginally, in France and Italy[1]. It includes online services (eGuichet), deliberation management (Plone Meeting) and web site template (CPSkin).

1. The project's international initiative, PloneGov, has never really taken off, illustrating the difficulty in sharing business developments between different countries, with different regulatory frameworks, even in Europe. The website is down.

Some challenges addressed by this case are common to any project: how to respond to users' needs, especially non IT specialists, how to share the costs fairly. Others are more specific to the public sector, e.g.: existence of supervisory authorities, pressure for self-financing developments, restrictive purchasing procedures that favor the established actors and solutions, a non-competitive environment that may favor cooperative actions.

The main result is that CommunesPlones, which was initiated as a classic FLOSS project, by and for the developers, has enabled municipalities to coordinate their efforts in order to create an original institutional framework that onboards non-IT users and maintains close control of the project by the users. They have done so by allocating their financial and human resources to one collectively owned semi-private structure. This has been to the detriment of direct in-code contributions.

This article, composed of four sections, analyses this choice. Section 2 discusses how FLOSS projects work and evolve. Section 3 presents the case and our methodology. Section 4 details the steps to build this institutional framework, whose model is discussed in the last conclusive section.

## 2 FLOSS to develop software solution for the public sector?

Von Hippel defined FLOSS as an organization, based on user-developers who possess both capacities and allocate some of their time to cooperatively develop a solution they need [2]. For these users, hereafter named innovative users, or "I-Users", in reference to von Hippel's concept of user-as-innovator, the possibility of making improvements is often less costly than waiting for the publisher of conventional private software to make appropriate upgrades. They also have an incentive to share these improvements: sharing bug feedback and corrections, modifications they have already made because they needed them, does not cost very much more. It is also a way to have these elements integrated and maintained in the software's future versions. Finally they benefit from others' efforts to meet the shared needs and to propose functionalities they may need but have not yet identified.

Public administrations have long been sensitive to these arguments [3], and have deployed generic open source software (operating systems, email servers, office automation, etc.) [4]. They have also developed dedicated FLOSS software (e.g. the Lutèce CMS developed by the City of Paris). The French non profit Adullact has been coordinating public authorities around FLOSS project development since 2002, with the goal to help them to regain control over how their needs are addressed, but also to better manage public IT spending (not paying twice for the same solution).

Despite these examples, mostly from large public administrations, benefiting from major pools of IT human resources, public authorities report difficulties in embracing FLOSS [5]. The concrete enactment of the two previously mentioned capacities remains challenging.

First, I-Users are not the only users of a FLOSS solution. Some users, who are able to assess the adequacy of a solution to their needs, may adopt the solution [6], expecting to benefit from a lower Total Cost of Ownership (TCO), beyond the zero license fee: the openness of the code makes it easier to fix bugs and adapt to user needs, but also ensure better compliance with standards. This facilitates interoperability and thus integration into the information system [7]. These so-called "F-Users", in reference to the notion of "frontier-users" put forward by Kogut and Metiu, are able to propose new ideas, but also to detect and document bugs in programs, and are very important in order for the FLOSS products to last [6]. However, their participation stresses the question of who allocates the resources needed to hear and develop their requirements.

To serve these users, and particularly the organizations, commercial IT providers have developed solutions based on one or more FLOSS products, in strategies described as "open-source"[2]. In this article, "FLOSS" refers to open software collective development projects – thus when I-Users participate significantly. "Open-source" refers to these companies' businesses based on FLOSS products [8], and more generally, on any software distributed under an open (source) license, even if one single actor produces all the code. These companies may help disseminate the FLOSS solutions, even to users without any IT skills (hereafter named "N-Users" for "no IT skills users"). It may help the project to reach a broader audience, but whether and how it provides it with extra resources is matter for discussion.

This questions the project's management, or "governance" [9]: the means put in place to manage, control, and coordinate potentially conflicting needs, proposed by different contributors, and how to finance them.

Studying a FLOSS project thus means questioning the participant's cost/benefit of involving in such governance processes, and the project's capacity to raise enough resources to endure.

We will now explain why CommunesPlone, launched as a FLOSS website project in 2005 by several I-User employees of Walloon municipalities, is an interesting case to study these questions.

2. See, for instance Markets and Markets' 2021 report.

## 3 Case studied and method

### 3.1 Choice of CommunePlone

The Walloon municipalities represent a market of 262 entities, more than half of which have less than ten thousand inhabitants. They have high and relatively similar IT needs, which makes it an interesting market. However, most have limited IT skills and budget. Because of this, most favor proximity solutions (chosen by a neighbor or proposed by a local provider). Also because it is easier for the supervising authority to select and control a small set of providers, the market is structured in local monopolies.

CommunesPlones was in competition with two initiatives supported by the supervisory authority, the Walloon Region, which is tasked with defining the IT strategies for the municipalities: Agoracités (web portal) and Qualicité (central purchasing and business software). The first was based on the hybrid software Jahia and was developed by a private company. The second relied on the open source software Alfresco for its specific developments, which were outsourced to private service providers. Despite this lack of "official" support, CommunesPlone's success (28 communes were partners in 2007, 75 by the end of 2008, 91 by the end of 2009, 120 by the end of 2010...) led to the termination of the Agoracités project, and to the merging of the management of the CommunesPlones and Qualicité projects into a new inter-communal structure organized on the basis of the CommunesPlones project (and employing the main CommunesPlone developers): IMIO.

CommunesPlones may thus shed light on the specific qualities of FLOSS over more conventional solutions for the public sector, but also on why such a structure was chosen – a private company owned by the public authorities which are its clients –, rather than more common FLOSS solutions, such as a foundation.

### 3.2 Method

To speak of addressing needs, and of the financial and non-financial resources mobilized to do so, and how they are funded (time allocated, money spent...), is to speak of a project's "business model" [11].

As for any projects, the needs, the resources, or the technology evolve over time. When the actors judge the current organization too unsatisfactory, they try to develop and implement one or more action plans to resolve the observed maladjustment. Doing so they change the organization, the governance of the project, its "business model" [12]. This calls for a longitudinal study of the project throughout its life-cycle identified by these moments of test and the solutions found to address them [12].

We collected data from public presentation of the project, including: several conference presentations by the CommunesPlone founders between 2005 and 2011; a feature in the 2012 Walloon local authorities association's magazine regrouping interviews of actors of the IMIO creation project; yearly management reports published by the IMIO from 2012. They helped understand the chronology of the project. Several academic reports were also commissioned by public authorities, on the pooling of IT developments [13], the rationalization of ongoing projects[3], and the involvement of public authorities in FLOSS projects [14]. They helped understand what were the difficulties to be addressed and the solutions proposed.

We completed these elements by interviewing two long-standing members of the project, during respectively 143 and 83 minutes. One has participated in the creation of IMIO and is still involved in it, the other is a former contributor to the project who has today a private consulting activity (on CommunesPlone and other public sector software solutions). The interviews helped clarify the previous points: the project timeline, the different challenges and how they were understood by the actors, but most of all the role of the specific institutional background: local politicians, local authorities, and companies. They also helped understand I-Users' view on the project's evolution.

We identified two periods of maladjustment/ readjustment, or "tests", that have led to the present model. We present the phases and their business model, and the elements which have pushed to their evolution using the business model canvas [11].

## 4 Results: From FLOSS to inter-municipal demand pooling project

### 4.1 The initial project-triggering problem

The project emerged in a quite classic way for a FLOSS project. In the mid-2000s, some municipalities found themselves in a situation of commercial dependence on local quasi-monopoly private service providers whose services were sometimes perceived as costly and of poor quality. The market for specialized municipal software had also been consolidated, after a major bankruptcy (AGD) and the gradual emergence of a dominant local player (CIVADIS).

3. An expert evaluation was committed when the inter-municipal organization was created that led to the institutionalization of CommunesPlone.

The small municipalities, with few to no IT resources had no choice but to accept this situation. Some of the biggest municipalities, such as Liège, had sufficient internal skills to develop their own, internal projects. In between, and in direct response to this risk of dependency, the IT specialists of several middle-size municipalities organized the informal pooling of resources to start a FLOSS project, CommunesPlone (Table 1). These municipalities had insufficient resources to do so independently, but enough to 1) see the issues related to the dependence on private service providers [14]: over-cost and poor addressing of needs; 2) collectively develop a competitive offer.

This initiative was possible for several institutional and organizational reasons. Despite the existence of regional supervision, the communes benefit from a high degree of decision-making autonomy. There was no additional spending (no outsourcing contracts as for the competing solutions). These IT specialists also enjoyed the trust of their superiors, who had sufficient IT resources –staff– to allow that some of their time was diverted from day-to-day production.

Two contingent elements allowed for the project's growth: the personal investment of two employees from two municipalities – one has embodied the project and become its "benevolent dictator", Joël Lambillotte –; the use of an already dynamic FLOSS project Plone, based on a diffused technology (the Python language), which provided a technological base and a community. They created an effective and cheaper solution than the competing ones, which started diffusing beyond the early adopters. This early success triggered the first test.

TABLE 1
CommunesPlone's initial business model.

| Key partners | Key activities | Value proposition | Customer relationships | Customer segments |
|---|---|---|---|---|
| Middle-size municipality IT services<br>Plone FLOSS project | In-house software development | Controlled & personalized software solution | The customers are the producers | I-Users in middle-size municipalities |
| | **Key resources**<br>I-User municipalities employees | | **Channels**<br>Internal development | |
| **Cost structure (time & money)**<br>Development time | | | **Revenue stream (time & money)**<br>Employees' time | |

TABLE 2
Advantages and disadvantages of the various FLOSS project business models. Adapted from [10]

| Models | Advantages | Disadvantages |
|---|---|---|
| Initial I-User project | Cost savings through pooling of human and financial resources. | Difficulties in funding an informal project and to scale beyond the I-Users. |
| Outsourcing to a foundation | Guarantee of commercial independence by integrating the project into a protective structure. | Unsuitable for small municipalities: no human resources to ensure deployment on an internal infrastructure. |
| Outsourcing to an open-source publisher | Provision of a packaged solution or, better still, SaaS, allowing development and hosting costs to be shared. | Risk of commercial and technical dependence from the provider. |
| Installation outsourcing to several service providers | Provision of services necessary for the installation and integration of applications. | Unaffordable for many small municipalities. Model that can complement one of the others but cannot work on its own. |

### 4.2 The success test

The ”voluntary” internal resources became insufficient to maintain the solution, but furthermore to install it in other municipalities, which did not have the human resources required. It became necessary to find funds to finance Plone and Python specialized providers. Then appeared the problem of the municipalities which did not have the financial resources to pay for the providers, hence to develop a more easy to deploy, standard solution. The project, and its participants explored, in two steps, several ”solutions” among those technically, institutionally, and organizationally accessible to them, beginning with the classic possible evolution for a FLOSS project, summarized in Table 2.

A short-term solution was explored, that could be seen as a way to finance installation outsourcing: delegation of personnel from the agency for administrative simplification (Easiwal), and responses to digitization project support programs from the association of Walloon municipalities.

It allowed participants to determine that the models based on outsourcing to a foundation, or to address the problem of installation only, did not provide a satisfactory solution to the problem of equipping small municipalities. They developed a hosting infrastructure, an early Software as a Service (SaaS) implementation. This increased the needs of funding, and of a more perennial, institutional solution.

The negotiation between the users (municipalities) and the supervising authority led to create an inter-municipal company (IMIO), which would ensure the development of the software as well as the administration of the hosting infrastructure (see Table 3). Both had reasons to prefer this solution over externalizing to a classic company. The authority saw it as a way to regain control over the entities it was supposed to control. The municipalities benefited from the possibility of contracting without going through a public market. However, this had unexpected impacts on the cooperative development. This was the second test.

### 4.3 The public open-source publisher test

The institutionalization of the project, enforced by its integration into IMIO, led to changes in the project's goal and management. On the demand side, IMIO concentrated on the ability to collect user needs, to pool them and to translate them into computer specifications, and development. This understanding of user needs is seen as a strength of the structure, which has succeeded in developing shared customer projects, organizing workshops to improve the appropriation of the software, and providing effective user support (ticket system). The development of the SaaS solution has ensured the dissemination of the solution to every municipalities, especially those with few resources.

IMIO's revenue stream has been progressively sustained, on the one hand, by a lesser dependence on public subsidies, and on the other hand, by members' contributions ensuring the self-financing of current projects and their hosting. This has limited the presence of private companies within the IMIO ecosystem to a few Python specialists providing an always smaller number of municipalities with solution deployment services. To maintain their business, these providers have turned to other actors (regions, communities, Brussels local authorities, etc.), outside IMIO's scope, that has favored the dissemination of the solution.

However, this has transformed the development side. IMIO has integrated several CommunesPlone key developers, and its quality expectations have rapidly exceeded the skills of the remaining occasional contributors. In practice, external developments were thereafter carried out by subcontracting contracts that are subject to public procurement. The creation of value through direct contribution from I-users has been relegated to the background.

The I-User community has been transformed into a club of loyal F-User clients, who collectively have sufficient weight to defend their interests in the face of the supervisory authority or the government. The FLOSS project, i.e., as we defined it, a project where the software is mainly developed by I-Users, appears dead. However, the project of serving the municipalities with a solution they control and which meets their needs has prevailed (see Table 4).

## 5 DISCUSSION AND CONCLUSION

This case sheds light on several points. A successful software project may mean the end of a FLOSS initiative, i.e. a

TABLE 3
CommunesPlone's intermediate business model.

| **Key partners** | **Key activities** | **Value proposition** | **Customer relationships** | **Customer segments** |
|---|---|---|---|---|
| Middle-size municipality IT services<br>Association representing municipalities<br>Plone FLOSS project | In-house software development<br>Structured software project management | Controlled & personalized software solution (I-Users)<br>Stable, better and cheaper solution (F-Users) | Customers-producers<br>Installation services | I-User municipalities: speeding up of the development, sharing of the costs<br>F-User municipalities: better solution (than private ones) for a better price |
| Private Plone/Python development service companies | **Key resources**<br>I-User Municipalities Employees<br>Key project developers | | **Channels**<br>Informal channels (municipality network)<br>Service companies | |
| **Cost structure (time & money)**<br>Development time<br>Outsourced feature development<br>Installation and maintenance services for F-Users | | **Revenue stream (time & money)**<br>Employees' time<br>Per-project non-recurrent financial support | | |

TABLE 4
CommunesPlone's current business model: IMIO's business model.

| **Key partners** | **Key activities** | **Value proposition** | **Customer relationships** | **Customer segments** |
|---|---|---|---|---|
| Municipalities involved (governance)<br>Plone community<br>Walloon Region | Software development<br>Software project management<br>Infrastructure administration | Maintained applications in SaaS mode<br>Central purchasing office<br>Needs specification and implementation | Inter-municipal<br>In-House relationship | Municipalities:<br>I-User municipalities: sharing of the costs, possibility to negotiate software evolution with a competent and listening provider;<br>F-User municipalities: a provider in capacity and willing to understand and implement their needs for a better price |
| Private service companies (secondary) | **Key resources**<br>"IMIO" Brand (trust)<br>Knowledge base: open software and customers' needs<br>CommunePlone Main developers | | **Channels**<br>Member meeting<br>Online customer support | N-User municipalities: better price<br>Other public authorities through private service companies (secondary) |
| **Cost structure (time & money)**<br>IMIO Employees<br>SaaS technical infrastructure | | | **Revenue stream (time & money)**<br>Municipalities yearly participation (support contracts)<br>Projects (new feature negotiated development) | |

project developed by its (I-)Users. This may appear "naturally" because mature software requires fewer technical contributions/adaptations, and is of less interest for I-Users, or because an actor becomes dominant and does not wish to finance the animation of and collaboration with the I-Users. The project may remain an open-source project, as the code is still protected by an open (source) license, but it is no longer cooperatively developed and managed (what we have called a FLOSS project). This stresses the question of the users' capacity to remain in control of the project.

While the CommunesPlone public FLOSS project started and evolved in a classic way for a FLOSS project, as summarized in Table 5, it has innovated through this new form of resource pooling by a trusted third party, IMIO, that allows the users to remain in control. Indeed, the IMIO structure is not a classic private operator, it is a public company created and owned by the users –the municipalities–, of different size by construction of the board, in order to carry out public service missions of municipal interest.

This has two consequences. First, the users (I-users and F-users) still have a strong say about their needs, even if they do not have the capacities to directly enforce them by commits to the project. The main goal of FLOSS is preserved: giving the control over the software agenda back to the users.

Second, because the users own the provider, and are involved in setting the prices for addressing their needs, they (F-users and N-users) are both made accountable for, and educated about, providing the necessary resources for the project to be sustained (software, even FLOSS, does not mean free of charge).

The open license remains important: it ensures the users, especially the I-users, an exit door if the quality of service deteriorates or the relationship becomes too locked-in by the structure. It also allows for peripheral private service providers to diffuse the offer beyond the inter-municipal user-owners.

Analyzing this arrangement in the lens of the two capacities [1] presented in the introduction, helps better understand in what it completes the FLOSS business models described in Table 2 (Figure 1).

The top two socio-technical arrangements require users

TABLE 5
Life cycle of the CommunesPlone project's Business Model (inspired by [12]).

| **Steps** | **Evolution of the business model** |
|---|---|
| Phase 1 (bus. model invention): | Search within several municipalities for a solution to create a municipal website (2005).<br>Creation of the CommunesPlone project with about ten participating communes (2006). Adoption of a FLOSS model in order to pool human resources and escape commercial dependency. |
| Implementation difficulty (test): | Difficulties in financing the developments of the FLOSS project. |
| Phase 2 (bus. model innovation): | 1) Support for the CommunesPlone project from several sponsors through funding or delegation of personnel (Easiwal, UVCW...). 2) Setup of a technical infrastructure to host and edit the solution. |
| Implementation difficulty (test): | Not a perennial solution to address multiple small municipalities' needs. |
| Phase 3 (final bus. model): | Creation of the IMIO inter-municipal organization (28 November 2011) to manage the project and scale up the infrastructure.<br>Transformation of the community into a customer club, following the disengagement of I-users, whose independence is preserved by the free software license and the publication of the source code, but furthermore by the control of the provider. |

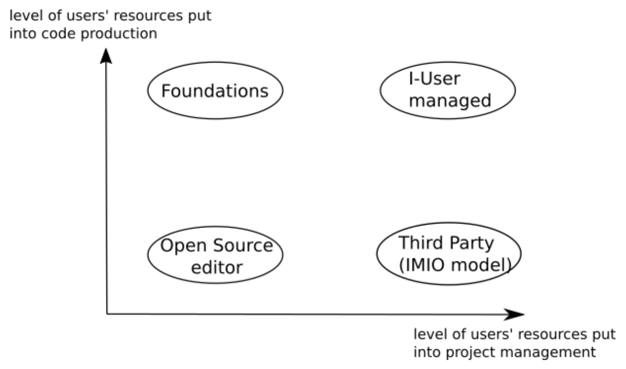
level of users' resources put into code production
Foundations
I-User managed
Open Source editor
Third Party (IMIO model)
level of users' resources put into project management

Fig. 1. Different technico-organizational solutions to control an open project, according to programming and software project management capacities

to invest in software production, and are thus reserved for I-Users. The organizations in which a central, autonomous editor exists, whether a private one or a foundation (left hand side of the figure), are less demanding for the users, but make it harder for them to control the project's governance.

I-Users always have the capacity to contribute in code (foundations) or to fork the project (open-source editors, third party, and foundation) if they are unsatisfied with the editor/project management. F-Users/N-Users do not. This is where having control over a third party which in turn controls the project may be interesting. This comes at extra cost for these users, both monetary (in order to pay the third party) and in effort (to participate in the strategic decisions and to learn to do so), but CommunesPlone/IMIO proves it potentially rewarding. Could this formula be transferable to other situations?

The institutional framework studied here is a ”solution” constrained by the technical and institutional resources actors could mobilize. It is not the best solution for any situation.

On the public side, some FLOSS projects managed by one public entity (such as Lutèce, as previously mentioned) seem closer to the private model, as one main user (Ville de Paris) controls the development. Adullact is probably closer to the third party model, but with a less committing framework for the users. Other types of Private non-for-profit open-source intermediate organizations also exist, such as the cooperative Coop It Easy, which proposed a Odoo ERP based platform, to remain in the Belgium context.

Software engineering needs more studies of these original arrangements to better understand how and when each open-source model is best to provide a long term solution to its various users, aligned with the project's governance and its users' values [15].

We hope this article's analytic canvas, based on the study of the provision of resources to produce the code and manage the project, on the one hand, and how non-I-Users are onboarded to contribute, on the other hand, will help do so.

## References

[1] S. K. Ethiraj, P. Kale, M. S. Krishnan, and J. V. Singh, “Where do capabilities come from and how do they matter? a study in the software services industry,” *Strategic Management Journal*, vol. 26, no. 1, pp. 25–45, 2015.

[2] E. von Hippel, “Innovation by user communities: Learning from open-source software,” *MIT Sloan management review*, vol. 42, no. 4, pp. 82–86, 2001.

[3] R. Deller and V. Guilloux, “Open source software movement in the french central and local government: a retrospective and prospective exploratory study,” *International Journal of Electronic Democracy*, vol. 1, no. 1, pp. 1–13, 2008.

[4] M. Cassell, “Why governments innovate: Adoption and implementation of open source software by four european cities,” *International public management Journal*, vol. 11, no. 2, pp. 193–213, 2008.

[5] M. Sowinska, B. Kudzmanaite, M. Shaikh, V. Devenyi, and C. Dussutour, “Guidelines for sustainable open source communities in the public sector: 5 key lessons: Practical guidelines for public sector representatives on how to join or establish and maintain sustainable open source communities,” in *14th International Conference on Theory and Practice of Electronic Governance*. ACM, 2021, pp. 491–493.

[6] B. M. Kogut and A. Metiu, “Open source software development and distributed innovation,” *Oxford Review of Economic Policy*, vol. 17, no. 2, pp. 248–264, Aug. 2001.

[7] F. Almeida, J. Oliveira, and J. Cruz, “Open standards and open source: enabling interoperability,” *International Journal of Software Engineering & Applications (IJSEA)*, vol. 2, no. 1, pp. 1–11, 2011.

[8] B. Fitzgerald, “The transformation of open source software,” *MIS Quarterly*, vol. 30, no. 3, pp. 587–598, 2006.

[9] M. L. Markus, “The governance of free/open source software projects: monolithic, multidimensional, or configurational?” *Journal of Management & Governance*, vol. 11, no. 2, pp. 151–163, 2007.

[10] A. Charleux and A. Mione, “Les business models de l'édition open source; le cas des logiciels,” *Finance Contrôle Stratégie*, no. NS-1, 2018.

[11] A. Osterwalder and Y. Pigneur, *Business model generation: a handbook for visionaries, game changers, and challengers*. John Wiley & Sons, 2010.

[12] S. M. Laudien and B. Daxböck, “Understanding the lifecycle of service firm business models: a qualitative-empirical analysis,” *R&D Management*, vol. 47, no. 3, pp. 473–483, 2017.

[13] A. Van Der Perre and D. de Roy, “Les pouvoirs locaux wallons face à la mutualisation informatique,” CRIDS, Université de Namur, Commissioned report, 2008.

[14] M. Zune, F. Horn, A. Vanheerswynghels, and D. Demazière, “L'implication des pouvoirs publics dans les projets de logiciels libres (projet osspa/open source software and public authorities), rapport pour la politique scientifique, programme société et avenir,” Centre de recherche METICES; Université Libre de Bruxelles, commissioned report, 2011. [Online]. Available: https://sites.uclouvain.be/libre/IMG/pdf/Rapport_OSSPA_TA11_024.pdf

[15] R. Viseur and A. Charleux, “Open Source Communities and Forks: A Rereading in the Light of Albert Hirschman's Writings,” in *17th IFIP International Conference on Open Source Systems (OSS)*, 2021, pp. 59–67.