FP7-SME-2008-2 – 243768

OPEN-SME

"Open-Source Software Reuse Service for SMEs"

Deliverable D2.6b
OPEN-SME business and cost models

Deliverable Type:	**PU***
Nature of the Deliverable:	**R****
Date:	**30/04/2012**
Distribution:	**WP2**
Code:	**OPEN-SME/UM-MERIT/WP2/D2.6**
Editor:	**UM-MERIT**
Contributors:	**AUTH**

* ***Deliverable Type:*** *PU= Public, RE= Restricted to a group specified by the Consortium, PP= Restricted to other program participants (including the Commission services), CO= Confidential, only for members of the Consortium (including the Commission services)*

** ***Nature of the Deliverable****: P= Prototype, R= Report, S= Specification, T= Tool, O= Other*

Abstract: This deliverable examines the conditions and forms under which OSS can be used for businesses. After a brief general introduction into the topic, a general overview of OSS ecosystems is provided. Thereafter, OSS business models and strategies are discussed, including results of a survey of OSS reuse practices in SMEs and the role of OSS within the portfolios of the SME-AGs and SMEs the OPEN-SME consortium. This section ends with practical recommendations for SMEs. Based on these insights and two workshops, an integrated business model for the commercial use of the OPEN-SME repository and tools has been created. Since costs of applying this business model depend highly on the concrete demand of the customer, the domain, the domain engineer, and the reuse engineer, a generic cost model for OSS reuse has been developed. The analysis concludes with an analysis of risks aligned with OSS reuse and ways to mitigate these risks.

Enosi Mihanikon Pliroforikis & Epikinonion Ellados	Project Coordinator	Greece
Drustvo za informacione sisteme I racunarske mreze-Informaciono drustvo Srbije	Partner	Serbia
Epistimoniko Techniko Epimelitirio Kyprou (Technical Chamber of Cyprus)	Partner	Cyprus
Teknikbyn Science Park Vasteras AB	Partner	Sweden
BITGEAR Wireless Design Services doo	Partner	Serbia
GNOMON Informatics SA	Partner	Greece
Maelardalens Hoegskola	Partner	Sweden
Teletel S.A. - Telecommunications and Information Technology	Partner	Greece
Aristotelio Panepistimio Thessalonikis	Partner	Greece
Universiteit Maastricht	Partner	Netherlands

This page has been intentionally left blank.

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

The notion of open source business models is widely determined by the well-known big commercial successes like Red Hat or Ubuntu. However, the big commercial success of OSS cannot be reduced to such big players and their business models. In contrast, we would like to argue that OSS is commercially successful because it can be used and integrated by everyone in many ways.

"Everyone" includes large companies, SMEs, Freelancers and single developers. The ways in which OSS can be utilized for business purposes vary considerably. For instance, a firm can focus completely on OSS or it can use OSS in order to complement its proprietary software, or it can sell services based on OSS, e.g. maintenance.

Against this background, another point that is often mentioned when OSS business models are discussed appears rather as a myth than fact: OSS business models do not always require an OSS community. In fact, the lion share of the OSS projects that exist do not have an active developer community. Only 8-10% of the OSS projects hosted on platforms like SourceForge, BerliOS or Savannah are stable and active, while the vast majority is inactive, with no single developer continuing work on these projects.[1] What is decisive for an open source business model is, in the first place, the availability of OSS code. How this code can be used to run a successful business model and whether or not an OSS community is needed depends on the capacities and the business strategy of the firm that runs this business model.

These insights, which will be detailed in this deliverable, have a strong impact on the key topic of the OPEN-SME project, the reusability of open source software and components. If only code that is supported and maintained by a community could be used for OSS business models, the amount of reusable code would be limited to only these 8-10% of OSS projects in the world that show a vivid community, and the huge remainder would either be lost for business opportunities or require from companies quite some efforts to build up such communities.

This deliverable is structured as follows: In the second chapter, OSS ecosystems are discussed in order to provide an overview of the environment and actors in which OSS business models operate. The third chapter provides an overview of OSS business models and business strategies based on an extensive literature review and distinguishes different types of OSS-related business strategies. The fourth chapter gives an overview of OSS business strategies and OSS reuse strategies in order to map the array of strategic options a firm has. The fifth chapter describes the OPEN-SME business model that has been drawn from these considerations and from two workshops that took place in Västerås (Sweden) and Athens (Greece) in January and February 2012. Next, a cost model for OSS reuse is described that serves as a guideline for businesses to calculate the costs related to a systematic and efficient OSS reuse practice as laid out in the RODE process developed by OPEN-SME partner AUTH. Finally, key conclusions with regard to the type, opportunities and challenges of the OPEN-SME business model are presented and discussed.

[1] This observation was made by the EU-funded FLOSSMETRICS project. See, for instance, Roberto Galoppini's blog: http://robertogaloppini.net/2007/08/23/estimating-the-number-of-active-and-stable-floss-projects/

2. SME-AGs and SMEs within the OSS Ecosystem and Value Network

The way open source software projects create value is significantly different from companies that create proprietary software. In the latter case, the value created is owned by a single company, whereas in the OSS case this form of ownership usually does not exist due to the OSS license under which the software is distributed. Open source projects are characterized by the fact that value is created for the network. Therefore, business models of companies involved in open source software projects cannot be analyzed isolated, they must be considered with regard to the companies links to other actors within – and sometimes without – the OSS project network [1].

As a consequence, in order to assess properly the opportunities search and evaluation tools for OSS and OSS components may offer to SME-AGs and SMEs, one has to take into account that these organizations are not acting independently of other organizations but are involved in a complex structure of collaborative and competitive relations to other players and customers. In the case of OSS, it appears meaningful to differentiate between the OSS ecosystem and the value network. Business ecosystems are defined in a broader sense than value networks, they comprise organizations as well as individuals, whereas value networks refer to organizations that contribute to the creation of value within a complex business process and usually emphasize a technical dimension in addition to relations between organizations and individuals.

For instance, Moore [2] characterizes business ecosystems as "an economic community supported by a foundation of interacting organizations and individuals—the organisms of the business world. The economic community produces goods and services of value to customers, who are themselves members of the ecosystem. The member organisms also include suppliers, lead producers, competitors, and other stakeholders." In this sense, SME-AGs are involved in a complex structure of relations to other SME-AGs, to their member organizations, to other firms that collaborate and / or compete with their member organizations, to customers and service providers, and to various OSS communities and their members.

Value networks are focusing the transition from linear value chains to complex structures of value creation. Normann & Ramirez [3], for instance, claim that "strategy is no longer a matter of positioning a fixed set of activities along a value chain. Increasingly, successful companies do not just add value, they reinvent it. Their focus of strategic analysis is not the company or even the industry but the value-creating system itself, within which different economic actors--suppliers, business partners, allies, customers--work together to co-produce value. Their key strategic task is the reconfiguration of roles and relationships among this constellation of actors in order to mobilise the creation of value in new forms and by new players. And their underlying strategic goal is to create an ever-improving fit between competencies and customers." A difference to business ecosystems is provided by a reference to the role of communication technologies [4].

The value network within OSS might be the same as the business ecosystem because in OSS consumers are able to modify the code (if no dual licensing or other restriction is hindering this) and thus to contribute to the value created, and SMEs and SME-AGs are able to connect with OSS communities through their technological infrastructure (e.g. SVN, mailing lists and the like). A more detailed discussion of OSS value networks is provided in chapter 4 of Deliverable 6.1a.

What is decisive for the functioning of both, ecosystems as well as the network of value creators, is that SME-AGs and SMEs that aim at offering search and evaluation tools must be capable to integrate in these structures and to (re-)configure them so that their offerings and demands are accepted by the other members of these structures. In the words of Normann & Ramirez [3]: "The new logic of value, and the dialogue between competencies and customers that it creates, presents every company with a stark choice: either re-configure its business system to take advantage of these trends or be reconfigured by more dynamic competitors."

The Reuse-Oriented Domain Engineering (RODE) process developed by OPEN-SME partner AUTH of this document portrays the value creation in OSS reuse as a chain of subsequent processes ranging from process definition to process configuration, domain analysis, domain design, domain implementation, and finally evolution. It must be noted that the value network perspective on the RODE process differs from the functional view of the software engineering perspective in so far that the described phases are not necessarily subsequent tasks of one and the same actor in the value network. For instance, the conceptual and analysis phases may have been carried out by a SME (within or without the value network) but this company may lack knowledge or capacities to execute the tasks related to domain implementation and evolution. In this case, an SME-AG might be able to perform these tasks as a (payable or free of charge) service to this SME.

SME-AGs become part of the OSS reuse value network if they manage to play, administer or assist at least one of the roles defined in the RODE process: re-users, re-use engineers, or certifiers.

Considering OSS reusing SMEs and their relation to SME-AGs in this context, our interviews[2] showed that only a minority of SMEs (three out of nine firms) sees a role for a SME-AG when it comes to OSS reuse. The reasons for this reluctance are very similar across all six companies that showed this attitude: they all trust very strongly in their own capacities and skills to evaluate what they need and their knowledge of existing code or components that come into consideration for reuse. All software houses that were interviewed pointed out their deep knowledge of the OSS community they are involved in and the software that is produced by this community. This knowledge is considered an asset that is ascribed only to people within the community and, even more, to employees of the SME. Somebody from outside the community would have difficulties to get this personal trust granted.

For instance, KDAB Deutschland, which is the company not organized in OSBA or other relevant SME-AGs, pointed out that there is not much sense in becoming member of a SME-AG or in SME-AGs offering search and evaluation tools and services for OSS. The striking point is, thereby, that KDAB Deutschland has quite a demand for OSS reuse and the aligned search and evaluation tasks. However, the company is used to perform these tasks internally because they have the required expertise and staff at hand to do so.

Another argument against using reuse services offered by a SME-AG is that SME-AGs are assumed to take too much time for these services. This argument is evidently closely correlated to the overall perception of superior in-house reuse expertise.

One company, Brox IT Solutions, was not totally reluctant towards the idea of SME-AGs offering code and component reuse services but pointed out that the financial capacity of many SME-AGs or OSS Foundations does not suffice to provide a proper and sustainable service. However, the company also sees a market opportunity for OSS reuse services, as the success of Black Duck Software illustrates that such offerings generate profit.

The opposite pole is provided by the three companies that can imagine to outsource software reuse services to a SME-AG or to purchase tools (or the right to use tools) for code and components analysis from SME-AGs. These SMEs are MyCRM, Millenux / Topalis and Boston Server and Storage Solutions.

[2] The companies that have been interviewed and analyzed for this document are: Agorum, Boston Server and Storage Solutions GmbH, Brox IT Solutions, C.A.P.E. IT GmbH, Jedox, KDAB Deutschland GmbH, Topalis / Millenux GmbH (Topalis is the mother company of Millenux), MyCRM and tarent AG. Altogether, these companies represent software houses, service providers and hardware vendors. In most cases, the SMEs offer a combination of software and services, whereby the share of software in the revenues ranges from 10% to 100%. Almost all SMEs we were talking to used OSS reuse repositories and tools, mostly Google Code Search, Black Duck and OSS project-specific or in-house repositories. Two companies have less than 10 employees, three have between 10 and 50 employees, three companies have between 50 and 100 employees and 1 company between 100 and 200 employees. All companies except for the hardware provider (Boston Server and Storage Solutions), reuse OSS regularly. Eight out of the nine SMEs that we have examined belong to the Open Source Business Alliance (OSBA - http://www.osb-alliance.com), which has recently been founded as a merger of the Linuxverband (LIVE) and the Linux Solutions Group (LISOG). While OSBA is an international association its members are mostly German companies, which has biased our interviews so far. One of the tasks of the future work in OPEN-SME is therefore to broaden the empirical basis of our observations and recommendations by including SMEs from other EU Member States.

Interestingly, the most pronounced demand in this regard was uttered by the hardware vendor (Boston Server and Storage Solutions), although this company does not practice OSS reuse. The motivation of the company to consider support for OSS reuse from a SME-AG results from the fact that effective reuse of OSS would probably help lowering the development time and prices for their hardware. Thus, hardware vendors could be an interesting market for reuse tools and the envisaged OSS reuse suite.

The other two SMEs are software and service providers, Millenux / Topalis also sells hardware. Both show, in principle, the same attitude towards OSS reuse services provided by SME-AGs as the other SMEs, i.e. they trust primarily in their in-house expertise. However, MyCRM would be willing to use reuse services from a SME-AG provided that quality and price are better than the reuse practice the company is used to, which is characterized by deep knowledge of the software it uses and develops and usage of software repositories (mainly Google code search) and product-specific code exchange platforms provided by OSS communities or other software companies. Overall, MyCRM sees a chance rather for specialized code and component reuse offerings than for generic platforms. Millenux / Topalis considers OSS reuse services through SME-AGs as an opportunity to save time of its own staff. Instead of assigning skilled software developers and software engineers code or component analysis tasks these tasks could be outsourced and the own staff could focus on the more strategic and valuable tasks like development of new software and services and customer care.

Two of the three companies that would use OSS reuse offerings from a SME-AG would be willing to pay for these services, but none was able to specify how much it would pay.

On the background of the prevalent assumption of superior in-house reuse expertise all SMEs tend to dedicate a very short period of time to OSS code and component search and analysis. For code and component search, the dedicated time varied between a couple of minutes and one work day, the time dedicated to analysis of code and components ranged between a few hours and one week. Thus, it must be assumed that as soon as a code or component search and evaluation would take more than one week SMEs would decide to develop the software they need in-house even if reusable code or components exist. The reason for these differences is that the scope and complexity as well as the importance of finding source code or components that can be reused vary from job to job. Given this uncertainty, the costs aligned with code / component search and evaluation were impossible to specify for all companies we have interviewed. Many companies found this question too abstract or theoretical.

Shifting from a rather representative function to a commercial service provider or supporter, as required from SME-AGs as distributors or vendors of OSS reuse tools and services, is apparently not an easy task for many SME-AGs. Our interviews with SMEs as well as with SME-AGs inside and outside the OPEN-SME project consortium indicate that SME-AGs are (self-)perceived to play a different role than SMEs. SME-AGs are often expected to perform political tasks, e.g. acting as lobbyists for the interests of their members, but they might be observed with some distrust whenever they start performing commercial activities that could potentially also be offered by their member organizations or that are considered to be inappropriate for the sector in which the SME-AG organizes its members.

Elmar Geese, chair of Linuxverband Deutschland (now merged with LISOG (Linux Solutions Group) to Open Source Business Alliance –OSBA)[3], confirmed that a SME-AG could find itself in a delicate situation with regard to its member organizations if it starts distributing software or software-related services, as such offerings would possibly set the SME-AG in competition with its member organizations. In addition, he pointed out that, like many SME-AGs, Linuxverband does not have the capacity to offer tools and services. The whole organization, so Mr. Geese, is designed to – and limited to - serve as an information provider and a promotion agency for open source software.

These insights suggest that SME-AGs have to first re-configure their relations to their closest partners within their ecosystem – their members - in order to become able to act as their members' and other companies' partner in their respective value networks.

[3] The content and outcomes of this interview reflect the situation at Linuxverband before the merger with Lisog. Statements made in the interview or referred to in this document must not be related to Lisog or Open Source Business Alliance.

However, this is not an impossible task. Normann & Ramirez [3] describe how the Association of Danish pharmacies had to react to new regulatory conditions and discovered in a situation of strong market pressures and economic threats two important assets: It's expertise in health care issues and it's huge network of contact points to the Danish population. The association started to reconfigure its activities and to focus on the provision of information rather than medical products. If it can be shown and communicated that member organizations and other companies, especially SMEs would benefit significantly from eased and non-competitive OSS search and evaluation tasks, SME-AGs would be in the perfect position to organize and / or offer such generic services.

Other SME-AGs that focus more on individuals, e.g. organizations of professionals, would probably face fewer difficulties to organize and offer such services and tools. Clarifying this question is subject to ongoing work in WP2.

The variety and diversity of SME-AGs and their different positioning within business ecosystems and value networks is evident even when only the SME-AGs in the OPEN-SME consortium are considered, which can be illustrated by the following three examples:

- Västerås Science Park (VSP) has broadened its company portfolio in the past and included also a University as strategic partner for fostering innovativeness and entrepreneurship in the region. The portfolio of the SME-AG is characterized by a core of 10 big companies in the field of automation, heavy robotics and technologies for independent life around which 160 SMEs are arranged. VSP is largely financed by public funding) from the municipality, the region and the EU) and employs 10 persons directly for the SME-AG. Its main service offerings to its members are networking (bringing together SMEs and large companies), management training and test beds for innovations. VSP characterizes itself as a platform and information provider and a service provider. It is not a software provider but plans to develop into this over the next 2 years. Finally, the members of VSP's portfolio are used to reusing OSS code and components.

- The Greek Association of Computer Engineers (EMHPEE) represents individual professionals as well as companies in its portfolio. Overall, EMHPEE represents 250 individual computer engineers and company representatives. Roughly, there are 20% freelancers and 80% company representatives, the latter are all from SMEs. The members' structure is very heterogeneous, including freelancers, academic researchers, and business representatives. The organization is governed by an elected management board, mainly consisting of public sector representatives. Staff, if needed, gets hired temporarily on a contract basis. The SME-AG is funded by the Technical Chamber of Greece and very low subscription fees of its members. The main offering of EMHPEE to its members is a magazine that is published every three months in order to inform about funding opportunities, results from scientific research and information of public interest, such as relevant law changes in the field. Training, workshops and institutionalized networking facilities do not belong to EMHPEE's core tasks. Since OSS does not play an equally important role in the Greek economy as in other European economies, OSS usage and reuse are rather untypical among EMHPEE's members. EMHPEE describes itself as an – unpaid - platform and information provider and as a service provider. The organization has not been active as a software provider until it got involved in the OPEN-SME project, therefore it does not aim at an important role in the software distribution within its network. The OPEN-SME suite is planned to be distributed via CD-ROMs.

- ISS, like EMHPEE, represents individual computer engineers and SMEs in the ICT sector. The organization has 1 full-time employed person which serves as a secretary in order to secure communication between the members and the board. The portfolio of ISS includes 49 members, of which 16 are companies and 33 are representatives of academia and research institutions. Although located in Serbia, ISS covers the whole area of the former Republic of Yugoslavia. The organization is funded through different activities, such as conferences, workshops, and projects. The services offered to ISS members are mainly workshops, seminars, symposiums, information days and the like, participation in local and EU projects. The main offering of ISS to its members is access to more than 5000 business contacts. Helping members to find business partners,

contractors and clients is therefore the key objective of ISS. ISS considers itself as a service provider; it does neither work as a platform and information provider nor as a software provider. The OSS market share in the region is below 1%. Overall, the biggest customer of ISS members is the public sector.

Below, after discussing OSS business strategies and models, we will present the main conclusions to be drawn from these differences with regard to recommendable OSS reuse business models for the SME-AGs.

3. OSS BUSINESS MODELS AND BUSINESS STRATEGIES

3.1 BUSINESS MODELS

3.1.1 DEFINITION

When OSS business models are considered, often the point of view is that of a comparison between proprietary software and OSS. In this regard, the typical argument is that proprietary software models are aiming at profits that derive directly from the software (i.e. the software is sold and revenues are based on IPR) whereas OSS business models are supposed to make profit rather from services (installation, maintenance, training) than from the software. In fact, the success of many OSS products with regard to diffusion rates and market shares has challenged traditional (proprietary) software business models and induced a shift towards more service-oriented business models in the proprietary software sector, too [5]. At the same time, a number of business and licensing models, such as the dual licensing model (e.g. used for MySQL), have been established in the market for OSS, allowing OSS enterprises to make profit from the software directly, too [6]. As a result, it would in fact be misleading to equate service-oriented software business models with OSS and software-related business models with proprietary software. The 451 Group [7] even claims that due to this convergence open source is not a business model at all but a model of software development and distribution. As we will show below, this conclusion is quite questionable because it ignores the impact of open source software on the structure of business models and on business processes and strategies. In fact, open source opens opportunities for business models and business strategies that cannot be tapped if a company only sells software and services related to proprietary software.

The term business model, though widely used within the academic debate as well as by practitioners, is not at all clear-cut. The term was introduced in the late 1950s but hardly used in publications until the 1990s, and only with the hype of the Internet it reached a first peak in 2000[8]. However, many authors mean completely different things when using the term [9] [10]. Amit & Zott's [11] finding that there is no commonly accepted or dominant theory or definition of business models holds still true, as Teece [12] still concludes that (business) economics did not provide a theoretical foundation of the concept of a business model (in this regard see also [13]). Teece [12] explains this shortcoming by the focus of economics on theoretical matters, whereas the business model concept aims at the solution of real world business problems and challenges.

In practice, business models are often conceptualized as a company specific feature, such as Apple's iTunes model, or as a generic principle to generate revenues, such as the freemium model. Academic conceptualisations of the concept 'business model' often focus on the interplay between product / service, the business actors, value creation and revenue sources (e.g. [14] [15] [16] [17]), others concentrate more on innovations and how to generate revenues from them [18], and others (e.g. [19]) emphasize the sort of transaction partners and channels (B2B, B2C, B2G, P2P), or the firm's position in the value chain and revenue [20] [21].

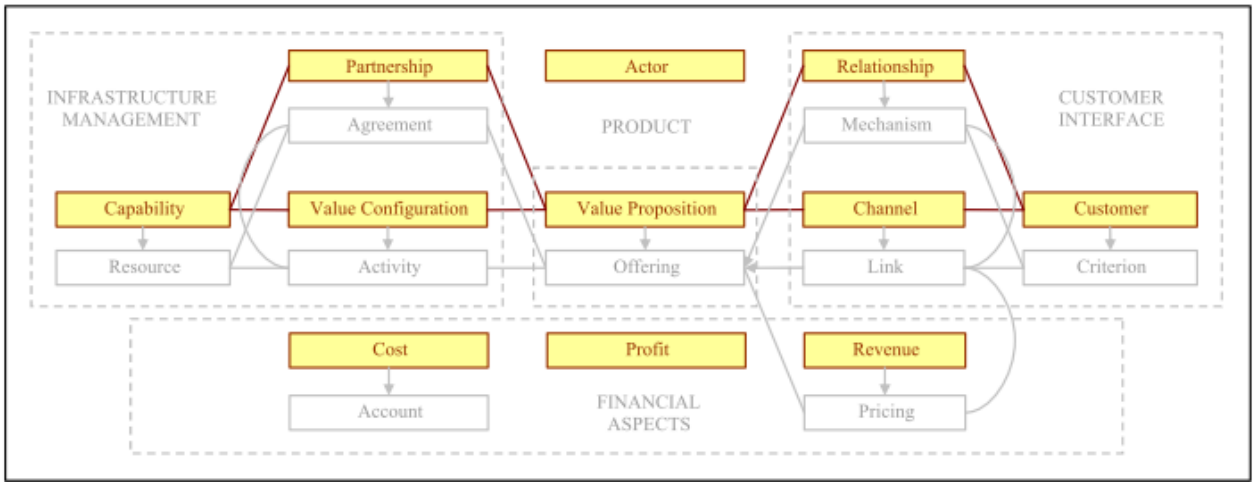
Finally, another group of authors try to systematically carve out components that are common to all business models, and to classify and analyze business models based on these components. Representatives of such an ontological approach are [11] [13] [22] [23] [24] [25].

From a practical point of view (and also with regard to the objectives of this proposal) it matters that there is no enterprise without a business model, regardless of whether or not a company can explicitly describe it [12] [26]. The reason behind this is that all enterprises strive for value creation and revenues and that their attempts to achieve this can be captured as their business model.

The strategic importance of business models results from the fact that businesses operate under changing market conditions, i.e. demand, competition, technologies etc. tend to change over time [12]. Business models provide a company with the means to adapt to these conditions individually: according to

Chesbrough [27], two companies selling the same product on the market would achieve different economic outcomes.

The problem of most of these definitions is that they emphasize different aspects that may be relevant for business models but do not systematically describe how business models are composed and how the different elements of business models are related and interact in order to generate revenue. Osterwalder [9] and Osterwalder & Pigneur [24] proposed a comprehensive ontology of business models that provided an overview of all business model components (Figure 1) and that serves as a starting point for our discussion of OSS business models in the context of open source software reuse.



Source: [9]

Figure 1: Business Model Components

With regard to OSS business models, the lack of structural analyses and ontologies appears even more distinct [6]. Extant classifications of OSS business models are usually based on the empirical analysis of business models found in companies (this case study approach was applied by, for instance, [28]) or in approaches combining literature and case study approaches [6] [29]. What they have in common is that, in contrast to Osterwalder's ontology, they do not start from a theoretical concept in order to create a taxonomy or ontology of OSS business models but conclude such a taxonomy or ontology from empirical observations. Such an approach tends to be affected significantly by the selection of cases and by the conceptual framework that is applied. In all studies on OSS business models that were examined for the purpose of this document, the conceptual framework refers to characteristics of OSS but not to generic categories that can be applied to business models in general, which would allow for a methodical comparison between business models in general and between open source software business models and proprietary software business models in particular.

For instance, Daffara [29] started his analysis from an initial set of variables that included, inter alia: "choice of licenses, product offering (whether a single version or multiple version of a software system are offered), services offered (divided into installation support, integration, training, consultancy, legal and technical certifications), type of contracts offered (subscriptions, licensing or per-incident) and metering form." Lakka et al. [6] followed a similar approach, as they derived four "main research issues" from the first round of their analyses: the kind of OSS license adopted, the offering or value of the OSS product or service, the OSS community, and the organization of production policy.

3.1.2 TYPES OF OSS BUSINESS MODELS – AN OVERVIEW FROM LITERATURE

Chang et al. [38] distinguish four models that secure sustainability of an OSS organisations

- community model
- subscription model
- commercial model
- central support model

In the community model is applied by, for instance, Apache and many other OSS projects that have established a foundation to secure their sustainability. The costs of sustaining the product or service are covered by building a community of users and industry partners who agree to cooperate on development work and maintenance [38].

In the subscription model, of which Red Hat is probably the best known example, requires users to pay subscription costs to a company in order to obtain maintenance and support [38].

The commercial model is also known under the terms 'dual license' or 'split license'. In this model the users have the choice between a free version of a software that gets no support and a version that must be paid but comes along with guaranteed support, maintenance and service models [38].

Finally, the central support model refers to a central body that provides robust releases and support for open source products that are of strategic importance to its community [38].

The list of OSS business model types that have been created around one of these sustainability models is extensive. For instance, Daffara [29] has identified six basic OSS business models (plus one remainder group), which only partly coincide with the five business models Dornan [30] has spotted in the OSS market. Summarizing these authors' findings, following OSS business models appear as typical:

- Product specialists
    - This business model ("Sell support services" in Dornan's terminology) refers to the traditional Linux model, as used by Red Hat, which, according to Dornan, is still part of most open-source business models. Basically, this is the classical OSS business model, where the code is completely open and revenues are achieved through services related to using, maintaining and adapting the software.
- Build (or run) hardware
    - This business model aims at making hardware more profitable through installing OSS on commodity components. Another form (the 'run' option) of this business model is to implement OSS on the firm's own infrastructure in order to produce software or services at lower costs.
- Proprietary components
    - This business model combines proprietary and open-source code, essentially holding back some functionality from what they release for free. Daffara [29] describes this model as distinguishing between basic OSS software and a commercial version that contains the basic OSS distribution plus additional of proprietary plug-ins. In the terminology of Lakka et al. [6], this business model relates to their term "added value editions".

- Dual licensing / twin licensing
    - Companies running this business model offer their clients the choice between open source and proprietary licensing models.
- Badgeware
    - Daffara [29] describes this business model as based on "a recent reinvention/extension of a previous license constraint, that is usually based on the Mozilla Public License with the addition of a "visibility constraint", the non-removability of visible trademarks or elements from a user interface. This allows the company to leverage trademark protection, and allows the original developers to receive recognition even if the software is resold through independent resellers."
- Platform provision
    - Daffara [29] use this category for "companies that provide selection, support, integration and services on a set of projects, collectively forming a tested and verified platform." They include Linux distributions in this category because they are (largely or completely) licensed under pure OSS licenses in order to maximize external contributions. The benefit for clients lies in "guaranteed quality, stability and reliability, and the certainty of support for business critical applications."
- Selection/consulting companies
    - Daffara [29] applies this category to companies that are rather consultancies and analysts than software developers. These companies provide consulting and selection/evaluation services on a wide range of projects. It should be noted that Daffara [29] assesses the impact of these companies on OSS communities as limited because "the evaluation results and the evaluation process are usually a proprietary asset."

Daffara [29] points out that there is a wide range of companies whose business models cannot be allocated to one of the above-mentioned categories because they combine (elements of) different business models. Of the many other labels that appear in literature in order to classify OSS business models and that cannot be allocated to one of the above-mentioned models (because they emerge across (some of) these types), only three appear meaningful to mention here, which have been discussed by Kudorfer et al. [28] in the context of the QUALIPSO project (www.qualipso.org). These business models are:

- Franchising model
    - This business model is based on a long term cooperative relationship between individual companies to offer specific products or services under explicit guidelines using the name, trademark and operational system of a franchise company.
- Subscription model
    - This model falls into the category of software as a service, where maintenance and updates are provided along the lifecycle of software. Customers are charged with monthly or annual fees for gaining access to the updates of an OSS product.
- Ecosystem enabler model
    - This business model is based on a company that plays a prominent role in establishing a de-facto standard platform, which results in the firms capability to dominate the related markets through control over the standards. Kudorfer et al. [28] refer to Sun Microsystems (in making JAVA OSS) and IBM (as enabler of the Eclipse platform) as examples of companies that apply this business model.

3.1.3 PROS AND CONS OF TYPICAL OSS BUSINESS MODELS

Chang et al. [38] compare following five types of OSS business models, which relate to a number of models described in the previous section, with regard to advantages and disadvantages (the following lists are quoted from Chang et al [38]):

- Support contracts
    - Advantages
        - Large organisations often require vendor support for their software and services, thus ensuring long-term sales and profits.
        - It provides a predictable and dependable revenue stream
        - Subscription renewal rates can be very high, thus ensuring a large number of clients and contracts.
        - It provides different levels of support for different organizational needs. This also provides users more options.
    - Disadvantages
        - A lot of customers feel there is no need to pay for support since the product is open source and plenty of free information is available.
        - It requires an existing base of customers to support, or it needs to ensure a large number of users already available.
        - It is easy for some organisations to clone an entire support architecture and services from an existing one, such as Oracle Unbreakable Linux.
- Split licensing
    - Advantages
        - Provides a high level of flexibility for users and organisation, which can retain both as an open source and commercialised operation.
        - It allows clients to use and customise the software for further sales without licensing restrictions
        - If clients' software sales include the software (such as MySQL), it increases the number of users and might increase potential sales.
    - Disadvantages
        - Some clients are confused with the boundary between commercial or GPL licence under the same product, particularly if they switch from using commercial support to OS support.
        - Any product or organisation in the entire sales chain, might be subject to licence and legal requirements if it is not guided or reviewed thoroughly.

- Community
    - Advantages
        - Backed up by a large community, community effort and product can become a mainstream such as Apache.
        - Being portable and functional on many products or platforms and widespread of world-of-mouth, it is presented and appealed to a wider range of users and organisations.
        - Can become a core component in a widely used product or platform, such as Apache HTTP.
    - Disadvantages
        - The leading developers or donators/investors may influence its development cycle and direction.
        - A lot of such organisations find it difficult to sustain and often request community donations
- Value-added closed source
    - Advantages
        - This is equivalent to commercialisation model where companies receive additional funds from share, investors' funds, sales commission, retailers and so on.
        - May generate much higher revenues if targeting the right market or products.
    - Disadvantages
        - If failing to impress users, clients and investors for some time, companies might fail to sustain themselves.
        - Certainly not OSS developers' favorites.
- Macro R&D Infrastructure
    - Advantages
        - Can easily attract funds from government, global partners or commercial organisations if they meet a specialised area where there are high demands for both R&D and investment.
        - Promote collaboration and partnership, and organisations may merge together to form a power-house in a specialized area to attract more expertise and funding.
        - Can create spin-offs and generate more revenues and useful research results, particularly for bioscience or medical or e-Science R&D projects.
    - Disadvantages
        - Sustainability model is under development and is influenced by investors (which might in conflict with initial roadmaps).
        - Need to seek funding with regular intervals, and can create a sense of instability and insecurity at those periods.

- Might be difficult to integrate academic theories and industrial perspective in some organisations.

3.2 OSS BUSINESS STRATEGIES

3.2.1 OVERVIEW

A business strategy describes the approach of a business to successfully compete with other businesses in a given market. According to Porter [39], "a company can outperform rivals only if it can establish a difference that it can preserve. It must deliver greater value to customers or create comparable value at lower cost, or do both."

In the case of OSS business models and strategies, OSS is considered to provide the means to achieve these advantages, given its inexpensiveness and availability. Therefore, we do not discuss all the facets of and different viewpoints towards business strategies but focus on the ways how OSS can be used to determine a business strategy.

The crucial point is that many of the business models described in section 3.1.2 and the support contracts model and the macro R&D infrastructure model described in section 3.1.3 perfectly capture how OSS is implemented by firms but fail, due to the convergence of OSS and proprietary software business models, to capture the specific impact of OSS on business practices (i.e. business models, business processes, and business strategies). To illustrate this: Almost all these models, but especially the franchise model, the subscription model, the ecosystem enabler model, platform provision, selection and consulting, and badgeware can be applied to purely proprietary software businesses, too. The only differences between the OSS business models and the proprietary software models would thus be the license under which the software is distributed and the openness of the code. Of course, these differences might be essential, but whether or not they are essential is not revealed at this level of analysis.

This reservation is supported by first results of the interviews carried out in WP2. Elmar Geese, CEO of the German open source software house tarent (www.tarent.de), a SME with more than 170 employees, and chair of the Linuxverband, an association of more than 30 German OSS companies, emphasized that in practice the differences between open source software business models and proprietary software business models are marginal. He points out, though, that OSS has "hundreds of advantages over proprietary software". However, Mr. Geese considers these related rather to the product, to the firm's strategic choices, and to the business processes than to the business model. According to Mr. Geese, a company can basically do four things with software, it can produce new software, it can recombine and integrate existing software, it can distribute it, and it can sell services based on or related to the software.[4] In this regard, it does not matter at all what sort of software – OSS or proprietary – the firm uses. What matters to Mr. Geese is that with OSS an enterprise can sell things to its clients that are not possible if it would base its business on proprietary software. Such value added that is created specifically through OSS is, for instance, insight (in the source code and thus in the functioning of the software) and control (e.g. by allowing to modify the code, extensively testing and evaluating the software, etc.).

Thus, another point that must be taken into account when the role of OSS for the economic opportunities and performance of a firm is considered is the firm's strategy and processes. This expansion of the perspective does not imply leaving the level of the business model. According to Drucker [31], every organization has a theory of the business, consisting of "... assumptions that shape the organization's behaviour, dictate its decisions about what to do and what not to do, and define what the organization considers meaningful results." The value creation strategy of a firm and the internal and external processes and relationships it organizes and it is involved in are, by nature, to a considerable degree reflected in its business model [25]. Osterwalder & Pigneur [32] even conceptualize the business model

[4] This typology resembles largely the categories of basic OSS business models introduced by Krishnamurthy (2003), who distinguished basically between software distributors (which would include Mr. Geese's category of software integrators), software producers, and third party service providers.

as the link between business strategy and business processes. This implies that similar or even equal business models – in terms of the categories depicted above - can pursue different strategies and actuate different strategic resources in order to achieve revenues. Consequently, within each type of OSS business model that was found by [6] [28] [29] [30] there is probably a very wide range of variation of the importance and impact of OSS for the business. With regard to our key question on reducing SME-AG's and SME's search, replacement and adaptation cost for OSS, this implies that in order to estimate these costs the specific OSS strategy within the potential clients' business models must be clarified. The underlying assumption of this proposition is that the costs for searching and evaluating a certain OSS or OSS component, though fix in general, might not translate into demand for the related services provided by an SME-AG or a SME.

The critical points of the business strategy with regard to OSS-related search, adaptation and replacement costs are as follows:

- OSS purpose: What role does OSS play for the firm's products and services?
    - Overall, there are three ways for companies to benefit from OSS: They can sell the software itself (e.g. based on a dual licensing business model), they can sell services for OSS, or they can implement it on their infrastructure (e.g. developing proprietary software on computers with Linux operating system, or selling services through channels that are completely based on OSS).
    - If OSS plays only a minor role, the company might not be willing to pay for OSS search and evaluation services. Instead, the company might prefer either abandoning this OSS or OSS component or waiting for an OSS community to come up with the software or the functionality it needs. In this case, offering search and evaluation services through a SME-AG or SME might only be competitive if performed in extremely short time and at very low costs. If OSS is critical for the firm, the readiness to pay for services and tools OPEN-SME is developing would probably increase significantly.
- What software and OSS capabilities does the firm have at its command?
    - If the firm is familiar with OSS and has enough expertise and capabilities in house in order to search, evaluate and / or modify OSS or OSS components internally it might be hard for an SME to get search and evaluation offers accepted because the firm trusts more in the established routines and in its own staff instead of a third party and tools. In this case, certified tools with certificates that are widely acknowledged by businesses could provide an option to overcome this obstacle.
- State of OSS community
    - If there is a large and vivid community around the OSS that is needed it might be possible to externalize search and evaluation tasks into the community. However, research [33] indicates that the willingness of community members to spend resources and effort on quality assurance that meets the demands of business is limited. If enterprises have made negative experiences with OSS communities with regard to quality assurance, this might fuel businesses' demand for the tools and services developed by OPEN-SME.
- How is the firm related to the OSS community?
    - If there is close collaboration between the firm and the community and the firm has a good reputation within the community, the firm might be able to hire community members for performing search and evaluation services. This is widespread practice among SMEs. Large companies might control or "own" the community, which provides the company the opportunity to define the standards of quality assurance according to its requirements, or to internalize this task completely. In this case, only if an SME-AG or

SME can demonstrate that it is faster and cheaper but at least as reliable as the company's own resources and routines. In this regard, certified quality of the tools and the processes needed to execute search and evaluation services with them would be helpful to lower entry barriers.

- Factors that influence OSS communities' capacities with regard to business requirements in general are
  - Size
  - Scope (geographical)
  - Expertise (number of skilled developers and 'project managers')
  - Degree of institutionalisation
  - Liveliness
  - Willingness to support businesses
  - Existing ties to firms

3.2.2 TYPES OF OSS BUSINESS STRATEGIES

Based on these considerations, we propose to distinguish between following OSS business strategies, which are not necessarily mutually exclusive:

- Full OSS collaboration
  - This strategy requires sharing and co-creation of OSS with other actors, namely an OSS community. Collaboration means a give and take to the benefit of all involved parties. For instance, the company might use existing open source software or components that has been developed by the community, and "repay" in terms of bug reports, bug fixes, patches, sponsoring events etc.
  - Examples: Red Hat
- OSS initiative
  - This strategy requires to initiate an OSS project and to establish a community around it. It is particularly applicable if a company owns software that can be released under an OSS license in order to attract more developers to contribute or to broaden the use base / number of application domains.
  - Control can be left to community (in this case it would be a hybrid strategy, combined with full OSS collaboration) or exercised by the company (in which case it would be a hybrid strategy combined with OSS takeover or OSS using).
  - Examples: Eclipse; Sun's decision to make the StarOffice code open source in 1999, resulting in the OpenOffice project.
- OSS takeover
  - This strategy means to take over an existing project and community and to control and steer the development. This strategy might be reasonable if the developed software is of strategic importance for the company's value propositions.
  - Example (though of a failure): Oracles takeover of OpenOffice.
- OSS fork

- This strategy means to create an own independent version of the software that is available from an OSS project or community. It appears recommendable if the software does not fully meet the requirements of the company or its clients or if the project is inactive and not further supported by a developer community.
- Forking leaves the community in control (otherwise it would be a OSS takeover or OSS using).
- Example: OpenPBX (fork of Asterisk).

- OSS using[5]
  - This strategy means to try to benefit from existing OSS code and community support without paying back to the community (e.g. in form of patches, sponsoring and the like).
  - No particular examples, but many companies use OSS without collaborating with the project and its community in any way.

The requirements of these OSS business strategies from the company and its business model are very diverse and will in many cases feature company specifics, therefore we will only roughly outline the key characteristics:

Full OSS collaboration:

The core capabilities demanded in this strategy are very strong OSS and community capabilities, i.e. deep knowledge of OSS and OSS communities, strong communication and management skills. Moreover, it requires an open partner network in which the notion of OSS communities is very important and where all business partners must be OSS compatible. The customer relationship is characterized by very close and stable relations that can be flexibly extended or limited for the purposes of co-creation. The value proposition can be software or services or both. The target customer can potentially be everyone, but given the expertise and capacities of the communities and the potential of this strategy for open innovation and co-creation, in many cases the target customer is supposedly an experienced user or co-developer. Costs can be saved through community support, and revenues can be increased through the community as well, as it can serve as multiplier.

OSS initiative:

Like for full OSS collaboration, very strong OSS and community capabilities are required core capabilities and an open partner network based on a vivid OSS community is decisive. Capacities to handle the community increase considerably if OSS initiative is linked with the aim to control and steer the project and its community. The customer relationship is characterized by very close and stable relations that can be flexibly extended or limited for the purposes of co-creation. The value proposition can be software or services or both. The target customer can potentially be everyone, but given the expertise and capacities of the communities and the potential of this strategy for open innovation and co-creation, in many cases the target customer is supposedly an experienced user or co-developer. Costs can be saved through community support, and revenues can be increased through the community as well, as it can serve as multiplier.

OSS takeover:

Like the two previous OSS strategies. Demand for community building and management capabilities is

[5] ‚Using' may not be the most appropriate term because OSS is used in all business strategies. We opted for the term because it indicates that OSS is considered only as a resource to be used for own purposes but not as something to which the company should contribute. An alternative term would have been 'OSS exploitation' but since OSS is produced in order to be used without an obligation to return anything to the community we found this term even more misleading.

higher than in full OSS collaboration and OSS initiative because an existing community is taken over and control over the project is the key objective of this strategy.

OSS fork:

The required core capabilities are strong software integration capabilities and maybe strong legal skills (in case of license issues), but since the benefits from a community shall be kept the company should also have capacities in community building and management, especially if the forking results from or leads to a conflict within the original OSS community. The partner network of the company may remain largely unaffected, though OSS activists within the community may have to change. Since a fork is usually driven by a demand for a different version of software, the value proposition of a company that forks an OSS project is surely software-oriented, but services based on the fork version can also be sold. Community support may decrease right in and after the process of forking but successful community building and management should be able to reconstitute the same or even a higher level of community support than in the original community. The customer relations should remain rather unaffected, though customers highly dedicated to the original version may disappear. Like the full OSS collaboration strategy, all channels can be used for the fork strategy. However, the benefit of the community might suffer a decrease in the process of forking. The specification of the target customer is not depending on this strategy, though if the demand for the fork version was driven by customers' needs it is likely that experienced users and co-developers provide an important target group of the company. If the original version of the software has caused a lot of potentially unnecessary work, e.g. adaptations of the software to client needs that were not supported in the trunk version of the software by the original community, cost can be saved through a better tailoring of the fork version. The established revenue streams should remain largely unaffected, and once the community is re-established and large enough the fork community can serve as multiplier.

OSS using:

Core capabilities required by this strategy are strong software integration capabilities and strong legal skills in order to avoid license infringements. Community skills are not at all required as this strategy aims at benefiting from the software without any interaction with the community. The partner network remains unaffected if the open source software or components do not demand significant changes from their business models. For the same reasons the relationships to customers should remain unaffected. The value proposition is not necessarily affected, as OSS using makes sense only if the company sells already OSS products or services or if the used OSS components play a rather marginal role for the company's value propositions. The company can keep its usual channels. Target customers can either be clients that are used to and familiar with OSS or persons / companies for which it does not play a role whether or not the software or service received from the OSS-using company is OSS. Costs can be reduced through replacing proprietary software through OSS, possibly also search, development and testing time can be saved through externalizing it to the OSS community. Revenue streams remain unaffected.

When OSS reuse is considered, next to the OSS strategy companies should have a OSS reuse strategy in place. As we have found out in our interviews, almost all SMEs have policies, guidelines or at least rules in place that make sure that OSS is reused and how this should be performed. The next section discusses and highlights important aspects of OSS reuse strategies, which depend on – and affect – the business model of a company and the company's position in business ecosystems and value networks.

3.3 OSS REUSE STRATEGIES

Both, suppliers of products and services and consumers, have to recognize that having a reuse strategy in place is important. Obviously, the reuse strategy is dependent on the business model and processes in place. For example, van Gurp et al. [34] compare practices for reuse in integration-oriented Software Product Lines (SPL) and large open source software projects. If an organization currently moves towards what they call an open compositional approach, it needs to change processes in order to allow for decision-making on the local level context of existing dependencies rather than on the global level, also

in terms of technical roadmap, requirements, and budget. Van Gurp et al. [34] conclude that software reuse comes along with loss of central control, which is something management must be willing to live with, or at least be aware of, if going for this approach. Further, going for open source reuse there is ample opportunities for products and services to assist in and manage the process.

Brown and Booch [35] argue that open source offers the opportunity for businesses to jump-start development and make the development more efficient both by reusing software, but also from best practices. These include the notion of broadening the notion of the project team, frequent releases, and greater collaboration across geographically dispersed teams. Brown and Booch [35] also find that open source innovations and approaches directly impact commercial software vendors. Some aspects can be built further on, others are in direct competition, and some enhances "coopetition". They focus on the aspect of the commercial software vendor who are vendors of software development methods and tools.

Aspects of special interest to commercial software vendors: open source business models, and the open source development process. Brown & Booch [35] argue that the tools used for open source software reuse must be well-matched to the development process. Understanding the open source development process offers insights into the requirements for any sets of services aimed at the software development community.

Businesses developing software, proprietary or not, are all candidates for reuse as they can consider to relicense their software. Making (parts of) the software open source may ease the market access for the product [35], and hence improve the product position and increase their revenue with add-on products and services. Brown & Booch [35] stress the importance of using appropriate tools. Both in terms of environment, synchronization and managing the process.

Further, reuse may have significant social benefits within an enterprise by connecting development groups, and within the OSS community by extending the develop networks and providing links between enterprises and the OSS community [36].

Another service currently available by Google Code Search tool is the possibility to search for publicly available source code by package, language, file name, class, function and license type (codesearch.google.com). This is a currently a beta service that helps the developer to find reusable code, and it is assumed that it will be supported by a revenue stream from advertisement as Google is doing for other parts of the company [36].

An example of another type of service a commercial software vendor could offer has been suggested to be a synthesis of the open source community for commercial software developers [35]. Getting an overview of the knowledge and assets available for reuse would help in the selection process in finding the best advice, knowledge, techniques and solutions. Parts of this process can be done by tools, where complementary services also have a role.

It is also interesting to consider Black Duck's best practices for managing Software IPs in detail, as this highlights multiple areas and tasks where there are opportunities for products and services in the reuse market. Some of these services are already offered by Black Duck Software, some are targeted at the internal organization but could possibly be outsourced as services and/or automated by tools.

An example of an OSS reuse strategy is provided by Pedersen [37], based on best practices of managing code reuse:

- Reuse existing components wherever appropriate
    - Component
        - Class, module, package, subsystem, file, etc
        - Internal or external
    - Rationale

        - Reduced development cost
        - Improved time-to-market
        - Reduced risk
        - Higher quality
    - Consider
        - Functionality and performance
        - Reliability and maturity
        - License obligations and risk
        - Sensitivity
- Track and control changes to internal components
    - When internal components are created or modified, capture the key metadata in your Configuration Management system
    - Track internal components deemed sensitive by a legal and business review
        - Embodies intellectual property critical to the organization
        - Risky in an important dimension
    - Rationale
        - Establishes and maintains the provenance of all internal components
        - Identifies and protects critical intellectual property
        - Prevents inadvertent violations of licenses, trademarks, patents, copyrights, and trade secrets
        - Constrains the propagation of risky software
- Control re-use of sensitive or external components
    - When usage of a component is first proposed it is useful to consider whether the use will be temporary or permanent, unmodified or modified.
        - Consider “form of use” (temporary vs. permanent, unmodified vs. modified, method of joining)
        - Consider license obligations
        - Formally review and approve
        - Record critical metadata (e.g. origin, version, license)
    - Rationale
        - Decisions are based on verifiable facts
        - Avoids last-minute surprises, guesswork, compromises, and risk-taking
        - Prevents loss of intellectual property
        - Facilitates remediation

        - Prevents schedule slippage from unplanned component replacement
        - Prevents damage to product or corporate brand

- Verify every build and release
    - At the creation of each project build or release assembly,
        - Identify all used but unapproved sensitive internal or external components or fragments
        - Identify all unapproved changes made to sensitive internal components or external components whose form of use requires their approval
        - Identify all changes made to external components whose form of use precludes changes
        - Find the root cause of any component misuse - take corrective action
        - Record metadata for all external components in the bill-of-materials
    - Rationale
        - Prompt discovery of component misuse, minimizing remediation costs
        - Enables demonstrable compliance with license obligations
        - Eliminates confusion caused by changes between project releases
- Review compliance at project phase transitions
    - At each major transition in a project's development process,
        - Verify no unapproved external or sensitive internal components or fragments have been added to the project
        - Verify no unapproved changes were made to sensitive internal components
        - Verify no unapproved or precluded changes were made to external components
        - Determine root cause of any component misuse, and take corrective action
        - Review license obligations of all external components used in the project, and ensure compliance, and compatibility with legal policies and financial objectives
    - Rationale
        - Assures prompt discovery of component misuse, minimizing remediation costs
        - Ensures that dynamic project objectives remain legally and financially compatible with all components used in the project
- Control component contribution and disposition
    - Before contributing any component or fragment to an open source project or transferring ownership to another party
        - determine whether the sensitivity of that component or fragment is an impediment to the contribution or transfer
        - verify the right to contribute or transfer every external component or fragment contained within the contributed component or fragment

  - Rationale
    - Prevents the inadvertent loss of internal intellectual property
    - Prevents license violation and attendant damages
    - Constrains the propagation of risky software
- Assess software components before acquisition
  - Before acquiring a significant interest in one or more software components,
    - identify all components that are internal, sensitive, and used in any active project
    - identify all external components used in any active project, and assess their license obligations with respect to compliance, business objectives, and legal policies
    - assess the impact of any required rework or change on cost, revenue, quality, etc.
  - Rationale
    - Prevents negative post-acquisition surprises

This is the outcome of the goals of reducing time and cost of development, being comprehensive, identifying issues early in the development cycle. Further, tracking IP compliance through the application life-cycle is also advocated, with regular validation and review. This can be automated and done by tools and offered as a service.

3.4 RECOMMENDATIONS FOR SME-AGS

Based on the analysis of the position and role of three SME-AGs that belong to the OPEN-SME project consortium in business ecosystems and OSS value networks, first recommendations of suitable OSS reuse business models for these (and similar) SME-AGs can be given.

Overall, VSP shows a very commercial orientation and must be considered as integral and important part of the business ecosystem in Västerås, in which OSS development and reuse are widespread. Conclusively, VSP plans to take over an active and commercial role in the distribution and implementation of the OSS reuse tools and services based on these tools by advancing itself into a software vendor for the OPEN-SME tools / suite. Business models developed for this sort of SME-AG should put the SME-AG in the centre of the model and strive to generate sustainable revenues directly for the SME-AG.

In contrast to VSP, EMHPEE and ISS are not part of their members' value network. However, their position in the business ecosystem of its members qualifies both organizations as distributors of the OPEN-SME suite. Further activities that imply playing a commercial role seem primarily to be limited by the governance structures and traditional tasks of the organizations and by the underdeveloped market for OSS in the two regions. Under such conditions, a suitable business model for a SME-AG should try to focus on commercial members of the SME-AG that are capable to play a leading role in the distribution, application and advancement of the OPEN-SME suite, while the SME-AG itself should rather serve as a non-commercial distribution and information platform. The latter may imply to advance the service offerings of the organization in the direction of training courses and networking activities. These activities could be organized in collaboration with member organizations. In fact, such activities take already place, but they are organized informally by the members. In the case of the OPEN-SME suite, institutionalized information events and training courses appear a more effective means to achieve an effective distribution and implementation of OSS reuse tools and services in the Greek economy.

In the case of ISS, the relative small size of the organization's portfolio creates a natural limit to the

distribution and exploitation of the OPEN-SME suite. Therefore, the business model should focus on a commercial partner that is capable to utilize ISS' huge network of business contacts in order to create a broader use base and thus ground for sustainable revenues from OSS reuse tools and services.

4. THE OPEN-SME BUSINESS MODEL FOR OSS REUSE

4.1 BACKGROUND

In order to create an integrated business model for OSS reuse through SMEs and SME-AGs, UNU-MERIT, AUTH and Västerås Science Park have organized two workshops with SMEs and SME-AGs. The first of these workshops took place on January 25-26, 2012, at Västerås Science Park (Sweden), the second one on February 16 at the premises of Teletel in Athens (Greece).

The design of the first workshop was characterized by two key objectives: (1) to familiarize the SMEs and SME-AGs with the business modelling approach developed by UNU-MERIT and largely based on the business model canvas developed by Osterwalder & Pigneur [24] and (2) to make the SMEs and SME-AGs acquainted with the OSS reuse tools and the Reuse-oriented Domain Engineering Process (RODE) developed by AUTH. To this end, the first day of the workshop was dedicated to presentations of the business model approach, the reuse tools and the RODE process. The second day was dedicated to strategic decisions and the possible interplay of actors in an integrated OSS reuse business model.

The purpose of the second workshop was to critically review the outcomes of the first workshop and to make decisions about roles and the interplay of the consortium partners in an integrated business model based on the OPEN-SME repository and tools.

In the following sections of this chapter we will subsequently develop this integrated OPEN-SME business model along the approach suggested by Osterwalder and Pigneur [24].[6]

4.2 GENERAL REMARKS

Being aware of the fact that the market introduction of a complex product like the OPEN-SME repository and tool needs time and a strategy, the partners have agreed to start the "OPEN-SME business" at a rather small scope, with VSP as key player for familiarizing, testing and implementing the OPEN-SME repository and tools in the robotics domain of the Science Park. In this initial phase, training and consultancy shall be provided by AUTH.

The roll-out, which provides the second phase, is intended to happen in different directions. The first one is collaboration with the SMEs and SME associations in the OPEN-SME consortium. To this end, VSP and the other OPEN-SME partners involved in the OPEN-SME business model will survey their members in order to find out to which degree and in which way OSS is used within their portfolios. Based on the survey results, good starting points for the roll-out of the OPEN-SME repository and tools can be identified. The second direction for the roll-out is provided by other Science Parks, as they have been identified as powerful multipliers with a perfectly matching portfolio of companies and domains in which the OPEN-SME repository and tools can be applied.

4.3 CUSTOMER SEGMENTS

A number of relevant customers have been identified. In the initial phase, the most important customers will be the VSP members, specifically those ones in the field of robotics. This approach has been chosen in order to familiarize with the OPEN-SME repository and tools in a controllable area. The robotics domain of VSP is particularly useful for the introduction and testing of the OPEN-SME tools and repository because these members of VSP have a lot of knowledge of OSS, so that the learning curve is assumed to be less steep than in other domains.

In the second phase, when VSP has accumulated enough knowledge about the OPEN-SME tools and

[6] For the "costs" component of Osterwalder and Pigneur's canvas we refer to chapter 5.

repositories, other Science Parks and Incubators will be approached. The International Association of Science Parks (IASP) has currently 388 members with overall 128,000 member companies,[7] thus providing a perfect platform for disseminating and applying the outcomes of OPEN-SME.

In a mid-term perspective SMEs (outside Science Parks) with a lack of reuse engineers (and maybe domain experts, too) shall find a possibility to directly receive OSS reuse services from the SME partners or other Science parks. Finally, in the long run, large companies shall find opportunities to receive large scale support (training, reuse service) for OSS reuse analyses.

The preconditions for successful offerings to SMEs and large companies is an effective and well-maintained website and a self-sustained OSS reuse community with expertise in a broad range of domains.

The value that can be created within the OPEN-SME business model serves, in the initial phase, three clusters within the VSP portfolio: robotics, smart grid, OSS. After the initial phase, following other actors will benefit from the value created by OPEN-SME

- wider VSP network
- other Science Parks
- OPEN-SME consortium
- Public sector
- SME clusters
- Software producing companies (not only software houses)
- Consulting companies
- Platform providers
- Quality assurance service providers (OSS & proprietary software)
- Individual developers / “geeks”
- OSS projects
- Academia (universities, students)

Besides robotics, other relevant domains for the OPEN-SME repository and tools are CRM, e-commerce, and banking, i.e. the OPEN-SME stakeholders will have to establish contact points to these domains and market the OPEN-SME outcomes in these areas.

For the geographical dimension of the roll-out strategy, the partners have decided to start on local scope, then develop markets on national and international scope. Multipliers, in this regard, are national contact points of the OPEN-SME partners and the International Association of Science Parks.

[7] http://www.iasp.ws/web/guest/facts-and-figures

4.4 VALUE PROPOSITIONS

Based on the capacities of the OPEN-SME repository and tools, a number of (bundles of) products and services can be offered to each customer segment, based on the examples provided by Osterwalder & Pigneur [24] (see Figure 2).

Figure 2: Elements contributing to value (Osterwalder & Pigneur)

The OPEN-SME tools & repository allow to offer analysis services and quality assurance. If services that are exclusively based on the tools are considered, OPEN-SME can offer help to solve legacy issues. The repository only allows offering components. Finally, the tools themselves can potentially be sold.

Next to the direct outcomes of the OPEN-SME project a number of services can be offered to the target groups, such as training and knowledge (initially by AUTH), support and consultancy (also initially by AUTH), domain engineering services, and brokerage.

A third group of offerings relates to building up a stack of expertise, as OPEN-SME allows generating experts in OSS reuse and reusable OSS components, in OSS (components) integration, expert users, and domain engineering experts.

Though there are a number of OSS reuse tools available, the unique selling point of the OPEN-SME approach is the combination of highly integrated reuse analysis tools on the one hand and the provision of a repository that allows direct access to reusable components with a so far unknown level of granularity. In this sense, newness and highly improved functionality are two core value propositions of OPEN-SME.

Another value proposition is performance, as the RODE process that is implied in the OPEN-SME approach towards OSS reusability allows improved process performance (systematic and efficient identification and testing of OSS code for reusability that goes far beyond what is possible today). Another feature resulting in improved performance is ease of identification of reusable software and its

classification (categories). In addition, the metrics applied or generated in the OPEN-SME approach will improve the identification and selection of best practices. Finally, the establishing of a code-reuse-oriented community will allow to externalize a number of tasks from companies / the OPEN-SME partners to other members of the community, which could particularly accelerate the growth of the number of components in the OPEN-SME repository. As a result, customized products (components, test results) will be available earlier than this is possible today, and components can be used systematically in OSS development, which is expected to significantly reduce the development time of new OSS products and services.

The latter point leads to a third value that can be offered to clients, which is customization. This is achieved through tailoring the RODE process to domain-specific and company-specific needs, which may include the modification of tools.

A fourth value to be offered through the OPEN-SME business model is the capacity to help companies that so far are not able to perform effective code reuse analyses to get this job done.

Overall, the partners intend to establish the OPEN-SME repository and tools as a brand. Given their newness and uniqueness, their qualification for a branding strategy is unquestionable. However, a comprehensive branding strategy depends on all partners' needs and capacities and has to be clarified and developed in a mid-term perspective (1-1.5 years). Challenges that have to be mastered in this regard are the name, which should reflect the core functionalities of the OPEN-SME repository and tools, a slogan, and a logo. "OPEN-SME" might not be appropriate, in this regard. However, other relevant cornerstones of a branding strategy have been identified: application fields / markets and the unique selling points are clarifies, as laid out above.

Overall, the branding strategy might benefit from applying Kano`s model of customer satisfaction [40]. Kano et al. distinguish 'attractive quality' from 'one-dimensional quality', 'must-be quality', 'indifferent quality' and 'reverse quality'.

Attractive quality provides satisfaction when achieved fully, but does not result in discontent when not fully achieved. One-dimensional quality provides satisfaction when fulfilled and discontent when not fulfilled. Must-be quality relates to features that are generally taken for granted when fulfilled but result in dissatisfaction when not fulfilled. For instance, a software program is expected to run without harming other programs or hardware after it is installed. When it fails to meet these demands customers will react extremely disappointed, though if the expectations are met they will not induce increased satisfaction. Indifferent quality refers to features that are perceived neither good nor bad, and they do not result in either customer satisfaction or discontent. Reverse quality refers to features that may result in high degrees of customer satisfaction in one or more group(s) of customers but a high level of discontent in other groups, due to differences in taste and preferences. For example, a product may be developed with a lot of extra features for 'geeks' that love sophisticated products with a large number of functionalities, and a basic version for those clients that feel deterred by too many functionalities.

The sixth value provided by OPEN-SME is design, as the implied focus on components eases and improves good software design.

Seventh, price is an important value to be offered by OPEN-SME, since tools and repository are OSS, which implies that the costs related to these elements are comparably low. However, it should be noted that the efficient usage of the repository and the tools requires high level expertise, which might result in relatively high prices for OPEN-SME services.

The latter point is however countered by the eights value OPEN-SME can offer, which is cost reduction. The outcomes of the OPEN-SME code reuse analyses are a broad set of well analyzed software and software components that are unlikely to produce in-house by most of the potential customers. This effect should outweigh expenses for high level expertise and overall result in lower production cost through

- larger supply with reusable code

- shorter development time
- ease of legacy management (for applications)
- less coding effort

However, as mentioned in the interviews, these cost reductions might not be perceived by customers (due to unawareness of costs aligned with no or bad code reuse). In addition, cost reduction might be countered by high learning costs and possibly high transaction costs (when introducing the RODE process in business processes)

The ninth value provided by OPEN-SME is risk reduction, as IPR issues become more transparent, extensive testing reduces the number of bugs in OS software and components, and the OSS reuse community and social network provides potentially a 24/7 service infrastructure. Especially SMEs will benefit from the latter

The tenth value provided by OPEN-SME is accessibility. OPEN-SME will ease the access to reusable software, components and test results through the Internet.

Finally, the eleventh value that will be offered with the OPEN-SME business model is convenience / usability, as the OPEN-SME tools and repository make it easier for firms and individuals to identify reusable code and components. Though the learning curve for handling the repository and tools effectively, it must be considered that so far OSS reusability analyses are performed by a rather eclectic trial and error approach that very likely overlooks many reusable components and does not provide comprehensive insights in the reusability features of the code under scrutiny. In this sense, the highly integrated tools and the OPEN-SME repository will turn out relevant information on reusable code in a faster and more comprehensive way in shorter time than the code analysis practices especially SMEs are use to so far.

These offerings help to solve a number of typical problems potential customers have when OSS code reuse is considered. In the first place, the OPEN-SME approach helps to structure the process of code reuse. In addition, OPEN-SME provides additional documentation of code that is not available otherwise. Furthermore, OPEN-SME provides customers with knowledge of software architecture that is lacking at the customer's side. The OPEN-SME tools and repository also help to increase scalability and to enter new markets. Another problem that can be solved by OPEN-SME is ease of training new employees and of knowledge transfer. Overall, OPEN-SME helps companies to focus on their core tasks while OSS code reusability analysis can be effectively outsourced.

SMEs benefit from OPEN-SME in particular through help in solving problems related to

- using and maintaining OSS efficiently
- time to market
- accessibility to code, high quality software, information about reusability of code
- tools
- skills
- knowledge

Individual developers will benefit through improvements of their

- status
- knowledge
- reputation

Against this background, following customer needs have been identified that can be satisfied by the OPEN-SME business model:

- Improvements of existing products
- Ease generation of new products
- Quality improvements
- Process optimization
- Decrease time to market
- Accelerated response to customer needs / requests
- Ease of support and maintenance

4.5 CHANNELS

There are three types of channels – distribution, communication and sales – that serve different purposes and play a role at different points in time. Figure 3 illustrates this for the OPEN-SME business model.

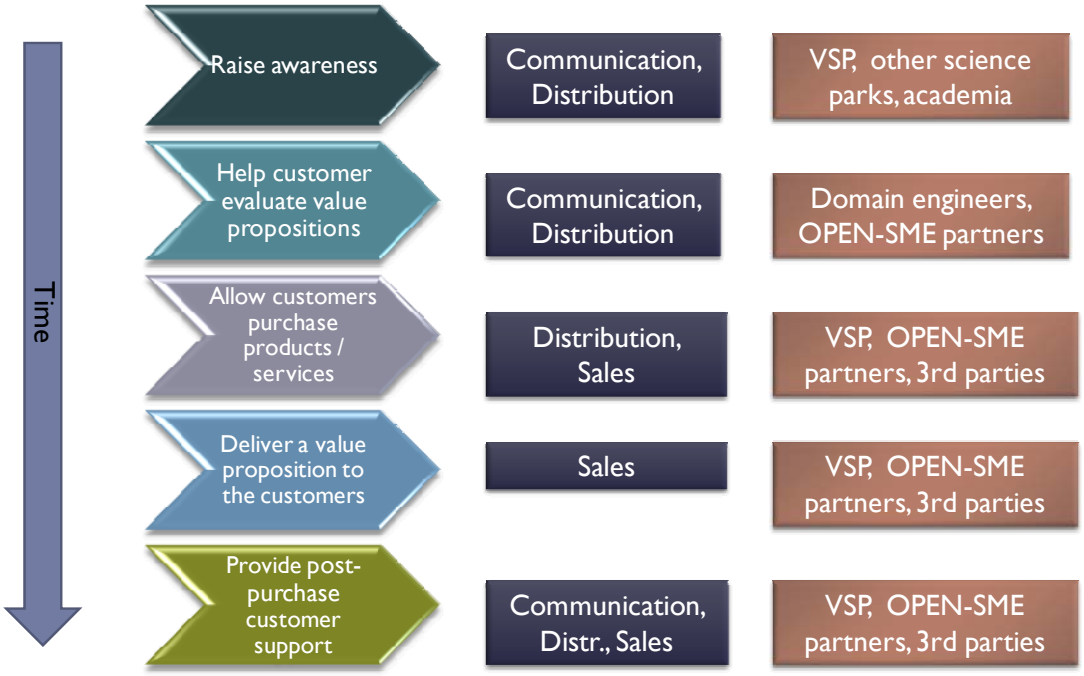


Figure 3: Purposes, phases and types of channels in the OPEN-SME business model

The OPEN-SME partners identified the following channels through which potential customers (target groups) presumably want to be reached.

- Internet (webpage, email)
    - Software communities
    - SME clusters / groups
    - Thematic forums
- Social media (Facebook, LinkedIn, Twitter etc.)
    - Registered “followers” from industry, academia and software communities
- Phone
    - Companies
    - Science Parks
    - EU networks
    - Industry Associations
- Face-to-face
    - VSP
- Teaching / courses
    - Academia
    - Industry associations / chambers of commerce
- Academia and industry collaboration
    - Master theses
    - Internships
- Events
    - Industry events
    - Software community events, e.g. FOSSDEM (fossdem.org)
    - Domain-specific events (e.g. conferences in the robotics area)

Since there is no similar service established within the partners of the SME consortium, it has to be evaluated which channels will be most effective. To this end, a number of measures has been discussed. During the test and pilots phase, events shall be broadcasted on the Internet. Challenges and opportunities shall be identified through benchmarking the success of different launches. Science Parks and SME clusters shall be attracted through direct contacts in existing networks. Showcases shall be created (prototypes, customer testimonials), and a downloads repository will be provided.

Measures that shall be taken particularly in the pre-market phase are presentations at GeekMeets and evaluation of feedback received from there, conferences in relevant industry domains, and a “Beta-version workshop” with early adopter champions from various companies (through Science Parks and SME clusters, partners’ networks).

In addition, EU networks and national and international events of / with other science parks and incubators shall be tapped. Finally, the partners decided to involve themselves in OSS associations and related events and in industry events, e.g. in the field of embedded systems (e.g. through ARTEMIS[8]).

These channels are not considered as means that work only in one way. Overall, the partners are interested in feedback on which components are used, characteristics of components' life-cycle, members' roles and flexibility, and how to establish continuous contact to users / customers / developers through active involvement.

Regarding the integration of existing channels, the focus of the discussion was laid on the infrastructures at VSP, as these are most decisive for the start of the business model and for the later roll-out. There is an established and well-tested communication strategy for VSP members that can be reused and integrated in a wider OSS reuse communication strategy. This includes the usage of VSP's CRM system, though this requires categorization of member types.

Based on VSP's infrastructure and the capacities of the OPEN-SME partners, following channels have to be integrated (integration is led by VSP):

- Established personal relations to key companies
- Personal contact points for distributing OPEN-SME outcomes
- Email, mobile apps, webpage

The integration of the OPEN-SME channels with customer routines shall be achieved through the creation of the "big picture" of OSS reuse. Invitations to cooperate in order to create this big picture shall be distributed to the target groups. Furthermore, a SME component pool shall be generated. The latter requires as a precondition the establishment of a critical mass of SMEs involved / interested in OSS reuse

[8] http://www.artemis.eu/

4.6 CUSTOMER RELATIONSHIPS

Figure 4 shows the different types of customer relationships that can be distinguished.

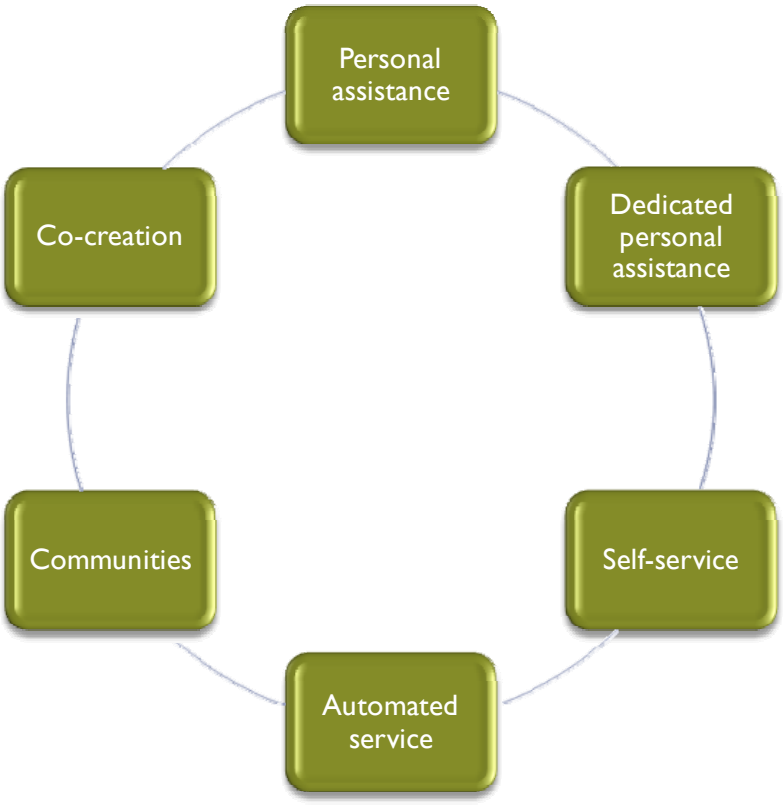


Figure 4: Types of customer relationships

The establishment of a self-sustained OSS reuse community is considered to be the key for all customer relations in the OPEN-SME business model. Regarding the types of relationships, the partners agreed that fully and semi-automated relationships should be avoided, as the complexity of the tasks probably does not allow for the level of standardization that would be necessary for these types of relationships. Within the community itself, self service relationships may be an option, as the level of expertise within the community should be high enough. However, the default setting for customer relationships should be personal relationships, maybe with dedicated personal assistance as a special case in domains or for large companies or SME clusters.

There are already a number of relationships established that can be used for the OPEN-SME business model: These relationships exist between

- VSP members
- other OPEN-SME SME AGs and their members
- OPEN-SME partners
- VSP members and OSS communities
- VSP and other Science Parks
- VSP / VSP members and industry associations
- VSP and government institutions

- VSP and academia
- OPEN-SME SME-AGs and industry associations
- OPEN-SME SME-AGs and government institutions
- OPEN-SME SME-AGs and academia

4.7 KEY ACTIVITIES

Key activities that must be performed in order to run the OPEN-SME business model successfully are twofold, on the one hand they have to help preparing the market for the OPEN-SME tools and repository and the services based thereof, on the other hand they have to secure and advance the value propositions offered to the target groups.

One key activity that is important in the initial phase is a survey / overview of OSS activities within the portfolio of the SME-AGs and SMEs of the OPEN-SME consortium. This survey would provide an initial overview of the markets for the OPEN-SME tools, repository and services and contact points for entering these markets-

Other activities related to market preparation are community building, the provision of experts and expertise, problem solving capacities (directly or through portfolio members), sharing of investment costs, organizing events and training (initially by VSP, either in Västerås or in Stockholm), and the dissemination to other Science Parks and SME clusters, industry associations and the like.

To the same end, key partners have to identify contact points in relevant domains, provide software components, testing, promotion (including academic and commercial publications, such as journal articles and whitepapers), and distribution.

Activities related to securing and advancing the value propositions are updates of existing software, software extensions, integration of additional functionalities in existing software, and certification services for special high quality software and components.

4.8 KEY PARTNERSHIPS

There are different types of key partnerships that serve different purposes, as illustrated in .

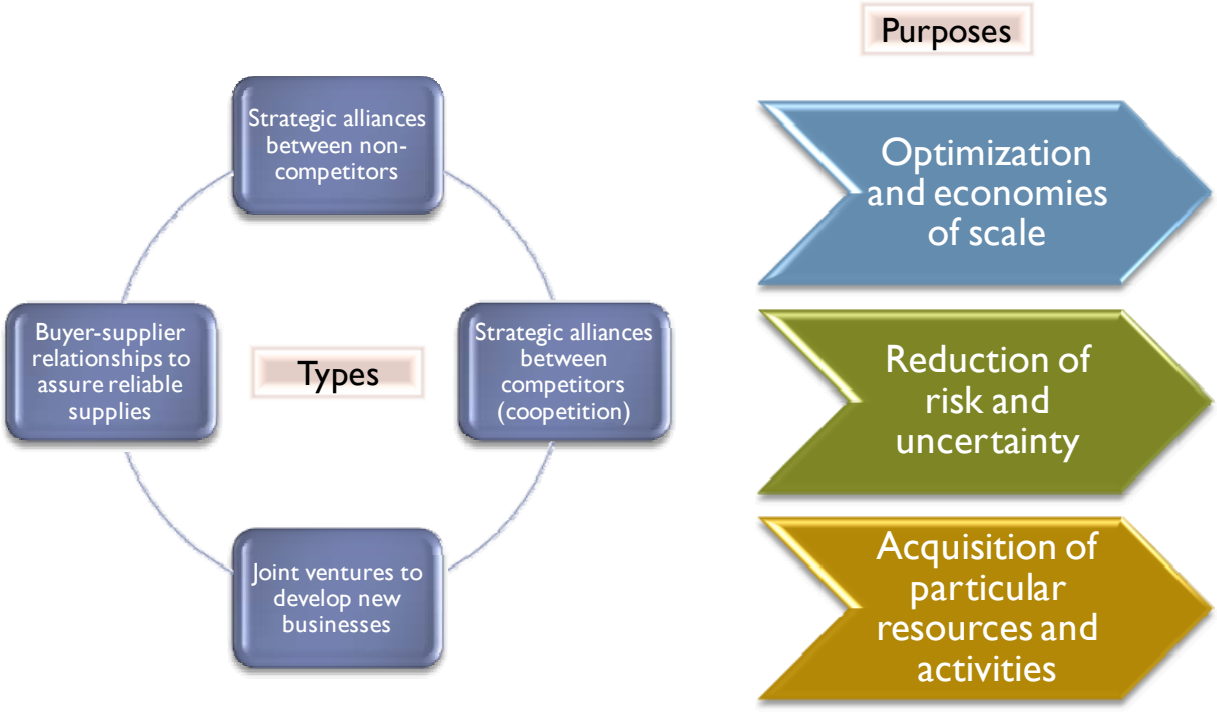


Figure 5: Types and purposes of key partnerships

The key partners in the OPEN-SME business model are, in the initial phase, the partners of the OPEN-SME consortium and the VSP member companies (especially in the field of robotics). These partnerships can at current be considered as informal (as not based on a contract) strategic alliances between non-competitors.

At a later stage, when a critical level of OSS reuse expertise has been built up at VSP and OPEN-SME consortium partners, additional contact points in relevant domains (which have to be identified by the partners), in particular other Science Parks have to be integrated in the business model as key partners. In this case, other forms of partnerships may be chosen, and the relationships might get formal (i.e. based on contracts).

A special key partner is academia, as academia does not strive for commercial revenues but plays a vital role with regard to quality assurance, branding, publications and promotion of OPEN-SME.

The key suppliers of the business model are, in the initial phase, the AUTH-team (reusability analysis, training), later the key suppliers will be part of a self-sustained community of SMEs, freelancers and volunteers, related to VSP members and other Science Parks, OPEN-SME partners and academia.

The key resources to be required from partners are

- Software components
- Trust building / branding capacities and efforts
- Manpower / expertise
- Networks / contact points

4.9 KEY RESOURCES

There are a number of key resources required by the OPEN-SME value propositions. In the first place, there is an essential need for domain experts, first in the field of robotics, later in other domains, too. In addition, hardware is needed for server and storage capacity. Cloud computing was considered to be an inexpensive and efficient and flexible option, in this regard. Other key resources are assistance in building the OSS reuse the community / network and clarifying IPR conditions (rights to OPEN-SME repository and tools).

In the introductory phase there is an "enabler" needed, i.e. initially one person in charge for introducing the OPEN-SME tools and repository at VSP. This person has most likely to be provided by AUTH.

Finally, a clearly defined timeframe and network in which the OPEN-SME tools and repository are intended to be applied in the initial phase and later roll-out is required.

Key resources required by the customer relationships are

- Clarification of target groups
- Identification of domains
- Businesses and contact points
- Network
- Branding (through existing distribution channels)
- Grassrooting / community building (as part of marketing)
- Regularly updated webpage with relevant information
- Timely information with regard to components etc.

Key resources required by the distribution channels are

- Survey of VSP companies
- Marketing capacities
- Mapping of target markets
- Branding experts
- Networks
- Identification of relevant events (industry events, academic events, policy events etc.)
- Contacting and coordination with other Science Parks, surveying their OSS capacities and needs
- Strategy: what to do in which order
- Financial resources
- Early adopter champions

With regard to the market introduction of the OPEN-SME repository and tools, for which the identification and approach of early adopters is extremely important, Mohan [41] warns that a blog post or a launch at a startup event or a press article will not suffice to succeed. He suggests "a disciplined 3-

step approach":

- Profiling and Identification (persona creation)
    - For B2B, 4 important characteristics to profiled and identify early adopters:
        - Location
        - Title of buyer (for the OPEN-SME business model, decision-makers for software development and software purchases are probably most relevant, but the survey should validate this)
        - Industry/domain (the survey has to identify the OSS-reuse-intensive domains)
        - Size of company (according to Mohan, mid-sized companies and a few large companies tend adopt new innovations faster compared to smaller companies)
    - For B2C , additional characteristics to consider are, inter alia, age, location, gender, monthly income among others.
- Interaction and Introduction - make an initial connect with early adopters through (one of) following three mechanisms (cited from [41]):
    - "Engagement online: Following them and posting thoughtful (real human) comments (not spam or robot messages) on twitter or their blog.
    - Events: Instead of presenting at a booth when your startup is not ready, demo your mock-up or early version to them at events (as an attendee) to get feedback.
    - Introductions from other early adopters. Early adopters know each other well and tend to be connected to each other well. They are usually open to sharing new, innovative ideas with other early adopters."
- Nurturing and Engagement – get feedback from early adopters and offer them to influence the product direction with the goal to categorize early adopters into 3 types and focus on making your champions successful with your product (following is cited from [41]):
    - "Champions: They like your product, think it solves a problem and are willing to provide feedback on what they would like, to make it better. Your goal should be to make these users the most happy with your service, be very responsive and introduce features they desire quickly. You can find them by looking at the # of times they return to use your service after the launch day.
    - Bandwagoners: They typically join since some other early adopter has joined who mentioned the product. They will come if the product is free, test it for an initial period, then will usually never show up until it is "more mainstream" or "many bugs have been worked out".
    - Naysayers: They have something negative to say about every new product, so while its best to ignore them, be thoughtful and respond to their feedback, but don't focus on them a lot. They will highlight many features that you currently don't have or plan to have. They are most likely to compare it to other solutions and in a negative light."

4.10 REVENUE STREAMS

Revenue streams can be generated in various ways, as illustrated in Figure 6.

Figure 6: Ways to generate revenues

Given the interview results it is obvious that customers are not easily willing to pay for OSS reuse analysis and services. However, the workshops have identified a number of values that appear attractive enough to be paid for by the target groups.

The first value in this regard is certification, as this service provides a sort of guarantee that the software or component does what it is supposed to do. The idea of the OPEN-SME partners is to provide a medium-level certification that can be issued based on extensive testing but without going through the time consuming procedure of strictly formal certification, like by ISO standards.

Another value that target groups are expected to pay for is tested components. Here, customers have to pay for the tests, not the components, as these are OSS.

Thirdly, extra documentation seems to be a value companies and freelancers would probably be more likely to pay for.

Premium models with extra information, exceeding the information generally provided to everyone, could also provide a value customers are willing to pay for.

Other such values are:

- Security
- Established and trusted brand
- Test and quality assurance

- Basis for demand of services: reference implementations and reputation
- Tools (if partners decide to sell tools)
- In SME clusters: additional service that can be provided to members' customers

Regarding what services and products potential customers (here: VSP members) are currently paying, it turned out that this applies to hiring of internal programmers, consultancy (to a limited amount), commodity software, and available components (very rarely). As a general rule, if a product or service does not serve the core business the willingness to pay is rather low. However, when problems arise or cost savings become evident the willingness to pay increases.

Regarding preferences of types of payment there was a strong agreement that one time payments have to be the default, as subscriptions and licenses are usually rejected by the potential customers.

5. REUSE COST MODELS

There exist in the literature numerous efforts in capturing the economic factors that can affect the success of a software reuse strategy. These approaches include cost-benefit analysis of software reuse, cost estimation, pricing criteria and identification of reuse support costs. Various authors [1], [3], [4], [5], [7] and [12] propose cost elements that should be considered when assessing software reuse.

Authors in [5] present how productivity gains can be quantified by the difference in development costs that stems from reusing existing assets versus developing custom assets from scratch. Quality gains can be quantified by the difference in maintenance costs that stems from reusing existing assets versus developing custom assets from scratch. Time-to-market gains can be quantified by the difference in development schedule that stems from reusing existing assets versus developing custom assets from scratch.

In [4] the categorization of the cost of software reuse as ***creation, development and maintenance*** costs; ***performance degradation*** costs, the ***risk of obsolete assets***; and ***security considerations*** is proposed. Nevertheless, the bulk of costs from implementing reuse are from the development and maintenance of reusable assets, tools and one or more reuse libraries.

In [12] authors characterize the reuse cost elements, as follows:

- Product Construction Costs, which are comprised of:
    - Asset Acquisition Cost (includes the cost of purchasing the asset from a source outside the product, as well as the effort invested in seeking the appropriate asset)
    - Asset Development Cost (includes the cost of the analysis, design and coding of new or modified artifacts, as well as the cost incurred by all the verification and validation activities performed directly on the asset)
    - Product Integration, Verification and Validation Cost (includes the cost of partial and full integrations, design reviews, subsystem and system testing, and acceptance testing)
- Core-Asset Construction Costs
    - Asset Acquisition Cost is the same type of cost as for product assets
    - Asset Development Cost is similar to the development cost of product asset development. Product integration, verification and validation costs do not apply, but for compound assets constructed in ways similar to the product, the cost of integration, verification and validation incurred until the core asset is ready for use may be considered entirely as this asset's development cost.
- Infrastructure Costs
    - Repository Establishment and Maintenance Cost (includes the cost of database analysis and design tool development or purchase, database administration, etc.).
    - Repository Storage and Cataloguing Cost (includes the cost of the effort needed for approving the artifacts for the repository and determining the metadata required in order to enable efficient search and retrieval of core assets in the catalogue).
    - Domain Analysis Cost (incurred by a set of activities needed to define the scope and the contents of candidate reuse assets, together with locating them in past products and making them available for the core-asset repository).

In [2], the generic costs of reuse may include those associated with ***selecting, understanding, modifying, certifying*** and ***maintaining reused software***, and can be employed in calculating the anticipated cost savings depending on numerous factors as:

- The development history of the software system (of which the artifact is a substantial portion).
- The cost of creating/maintaining a reuse library of software artifacts.
- The percentage of the system that is created using existing software artifacts.
- The percentage of change in each software artifact being reused.
- The life-cycle model used in the software development process.

There are numerous models assisting in cost assessment of a software reuse strategy. According to [7] there are three most common types of reuse economic models:

- *Cost avoidance models* calculating the amount of money saved from reuse in comparison to the amount spend for building the system from scratch.
- *Return-On-Investment (ROI) models* analyzing the net benefit of reuse, presenting benefits in the form of effort or savings. Extensions of this work currently focus on the time value of money for projects that extend over longer durations, taking into account the associated market and operation risks. The application of Real Options Analysis in the context of software reuse is a breakthrough towards this direction [13].
- *Cost-benefit models* providing assistance in analyzing reuse problems by highlighting all of the cost factors influencing reuse, sums the values for the appropriate factors and make judgments based on the model outcomes.

Literature contains a plethora of available reuse cost models, we present the most representative based on the needs.

5.1.1 REUSE COST MODELS IN DETAIL

In [8] author presents a basic reuse cost model developed by the Software Productivity Consortium (SPC).

$$COST = (1 - R) + R\left(b + \frac{E}{N}\right)$$

where,

COST	=	Cost of software development
b	=	Relative cost to reuse software (b = 1 for new software)
R	=	Proportion of reused code in the product ($R \leq 1$)
E	=	Relative cost of developing a reusable asset ($E > 1$)
N	=	Number of reuses over which the asset development costs will be amortized

In [4], detailed comparisons are provided between 7 reuse economic models using a common lexicon that he developed. The work proposes a Risk-Based Software Reuse Model as follows:

$$NPV = \sum_{k=0}^{M} \frac{\left[Cc_{wrp_k} - Cc_{rp_k} + P_k - Cp_{r_k}\right]\rho_k}{(1+i)^k}$$

Consumer

Cc_{wrp}	Cost to consumer to create product/system without reuse
Cc_{rp}	Cost to consumer to create product/system with reuse

Producer

Cp_r	Cost to producer to create an asset for reuse

Profit

P	Profit from increased revenues enabled by reuse

Risk

P	Probability of receiving cash flow

Time Value

I	Interest rate by which cash flows are discounted
M	Number of time periods under consideration

Output

NPV	Net Present Value

An extension of the work incorporates the cost of software debugging, under the perception that software is economically justified when it does not exceed the expected loss due to failure of the software. The break-even point (when the cost of debugging equals the expected loss due to failure) when the code component is used in one product is:

$$C_d = Epc_f$$

where,

C_d	=	Cost of debugging
t	=	Anticipated lifetime of a software product (or combined lifetimes of all copies of a product)
f	=	Relative frequency with which a code component is executed
E = tf	=	Number of times the code will be executed
p	=	Probability that a code component contains a fault
r = fp	=	Reliability; or the probability of code causing a product failure
c_f	=	Cost per failure of a code component

If the code component has the potential to be reused in N products, then the total number of executions of the component would be:

$$E_{total} = \sum_{i=1}^{N} E_i$$

The breakeven point when the code component is used in N products is, then:

$$C_d N = E_{total} pc_f$$

Since $E_{total} \geq E$, it must be true that $C_d N \geq C_d$. The more a piece of code is executed, the greater the expenses for debugging the code that can be justified since it is amortized over several uses. Reuse affords more debugging, which decreases "p", the probability that the code component contains a defect, and thereby improves "r", the reliability of the product.

In [6] and [11] a number of reuse cost measurements that quantify *Reuse Cost Avoidance* (RCA), *Project ROI*, and *Corporate ROI* is analyzed.

5.1.1.1 REUSE COST AVOIDANCE (RCA)

The **(RCA)** measurement quantifies the financial benefit of reuse. This is a particularly important metric because it shows the tremendous ROI potential of reuse. Since RCA is a key metric for performing ROI analysis of reuse programs, RCA helps with the insertion of reuse technology. The basic formulae for computing RCA are:

$$RCA = DCA + SCA$$

where,

DCA = Development Cost Avoidance (in dollars) and is calculated as:

$$DCA = RSI * (1-RCR) * NCC$$

where,

RSI = Reused Source Instructions (in thousands)
RCR = Relative Cost of Reuse (in decimal)
NCC = New Code Cost (in dollars per thousand source instructions)
SCA = Service Cost Avoidance (in dollars) and is calculated as:

$$SCA = RSI * (SDER) * (SDRC)$$

where,

RSI = Reused Source Instructions (in thousands)

SDER = Software Development Error Rate is the average number of total valid unique program analysis reports (TVUA) per thousand source instructions

SDRC = Software Development Repair Cost is the historical average cost per TVUA (in dollars)

5.1.1.2 PROJECT RETURN ON INVESTMENT (ROI)

The **(ROI)** is defined as:

$$Project\ ROI = RCA + ORCA – ADC$$

where,

ROI = Return on Investment that would occur in infinite time (in dollars)
RCA = Reuse Cost Avoidance (see above) for the initiating project (in dollars)
ORCA = Reuse Cost Avoidance for other projects benefiting from the reusable code written by the initiating project (in dollars)
ADC = Additional Development Cost of writing reusable code to the initiating project (in dollars)

and,

$$ORCA = \sum_{i=1}^{n} SIRBO_i * (1 - RCR_i) * NCC_i + \sum_{i=1}^{n} SIRBO_i * SDER_i * SDRC_i$$

where,

ORCA = As described above
SIRBO = Source Instructions Used by Others (in thousands) for each other project
RCR = As described above (in decimal) for each other project
NCC = As described above (in dollars per thousand source instructions) for each other project
SDER = As described above (per thousand source instruction) for each other project
SDRC = As described above (in dollars) for each other project

and,

ADC = (RCWR – 1) * CWRO * NCC

where,

RCWR = Relative Cost of Writing Reuse (number $\geq$ 1.0)
CWRO = Code Written for Reuse by Others (in thousands)
NCC = As described above

In [6] author defines the Relative Cost of Writing for Reuse (RCWR) factor as the ratio of the cost of developing a reusable asset divided by the cost of developing an equivalent custom-tailored asset.

5.1.1.3 CORPORATE RETURN ON INVESTMENT

The **(CRI)** measurement is based on the most common way to express ROI at this level, i.e., the internal rate of return (IRR) method, where the discount rate "k" is chosen such that:

$$C_0 = \sum_{i=1}^{n} \frac{(R_i - C_i)}{(1+k)^i}$$

where,

C_0 = Corporate reuse start-up costs (in dollars)
R_i = Revenue (savings) in year "i" (in dollars)
C_i = Costs in year "i" (in dollars)
N = Number of years for which revenues are to be considered
K = Discount rate (in decimal)

The Software Productivity Consortium (SPC) in [9] has also developed and published a model for calculating ROI from software reuse as:

$$ROI = \left[\frac{N * E * (C_{VN} - C_{VR})}{C_{DE}} - 1 \right] * 100$$

where,

N	=	Expected number of applications (versions, products, etc.) to be produced by the organization
E	=	Efficiency of the library infrastructure; the ratio of the amount of reused code in an application system to the available reusable code
C_{VN}	=	Unit cost per code size of new code developed for an application system
C_{VR}	=	Unit cost per code size of reusing code from the reuse library in an application system
C_{DE}	=	Unit cost per code size of the investment in the reuse program

The Reuse-Driven Software Processes Guidebook [10] presents a number of additional reuse economic models that can be used to determine ROI and the effect of reuse on software quality and productivity.

5.2 THE OPEN-SME COST MODEL

As determined in the RODE process, we make two broad categories of activities, carried out by the reuse engineer, namely non-repetitive and repetitive. Non-repetitive are those activities that will be performed only once in the process lifecycle, while repetitive are those, that will be performed during the evolution of the reuse repository assets. We make this classification in order to ensure that the right costs should be calculated when pricing a component.

In a same manner to the activities, we identify two general domain engineering scenarios. These are the application of the Domain engineering process for a requested component, and the Domain engineering process of arbitrary component. Again the logic behind this classification is to avoid duplicate calculations of costs of the adaptation activities incurred in a given component.

We adopt the notions of relative cost of engineering activities, Number of reuses, and the "Proportion of domain engineering activities performed" from the general cost model proposed by [8] developed by the Software Productivity Consortium (SPC).

We define the Total Cost of Process (TCP) through the following equation:

$$\text{TCP} = \sum_{i=1}^{i}[(b_i * Lc_i) + A_i] + (\frac{1-R}{N}) * \sum_{j=1}^{j}[(b_j * Lc_j) + A_j]$$

where,

TCP	=	Total Cost of Process
B	=	Relative cost of engineering activities - represents the fraction of cost per activity. The sum of all relative costs should be equal to 1. (b =1 for new component implying that all activities are required).
Lc	=	Labour cost for Reuse Engineer
A	=	Activities additional overheads
R	=	Proportion of adaptation activities requested ($R \leq 1$)
N	=	Number of reuses over which the asset development costs will be amortized
i	=	the range of additional activities (requested)
j	=	the range of all activities minus the i activities

It should be noted here, that the ***b*** factor is estimation and thus, can be either provided through simulations (Monte-Carlo simulation) or by regression analysis (when the mass of historical data from the performed activities are adequate enough to provide efficient results).

Based on the defined the Cost model, we infer the pricing model for the domain engineering activities performed on the requested component. We introduce the notion of "*probability for reuse*" (p<1), which denotes the probability for a given component residing in COMPARE, to be reused in the future. This estimation should be iteratively corrected upon component's reuse.

The resulting formula for pricing the performed activities is as follows:

$$SC = \left((1-p)\sum_{i=1}^{i}[(b_i * Lc_i) + A_{i]}] + (\frac{1-R}{N}) * \sum_{j=1}^{j}[(b_j * Lc_j) + A_{j]}]\right) + K$$

where,

SC	=	Service Cost to be paid by the SME
p	=	probability of component reuse.
K	=	SME_AG overhead costs, service commission, etc

We do not intent to treat the proposed pricing model as panacea every time a request for component is made to an SME-AG. We acknowledge the fact that existing ethics and pricing policies as well as unique business propositions may exist inside the SME-AG value chain that may constitute different/alternate pricing policies to be adopted.

Finally a brief reflection on how this OPEN-SME cost model relates to the Reuse-Oriented Domain Engineering (RODE). The table below maps the RODE process phases to the (primary) roles that have been identified.

RODE Process	**Identified roles**
Process Definition	Reuse Engineer
Process Configuration	Reuse Engineer
Domain Analysis	Reuse Engineer
Domain Design	Reuse Engineer
Domain Implementation	Reuse Engineer, Certifier, Reuser (optional)
Evolution	Reuse Engineer, Certifier, Reuser. (optional, in case of asset order)

Table 1: RODE Processes and Roles

As pointed out in the section on value networks, these roles can more or less be freely allocated to partners in the business ecosystem or value network. If the roles and thus the work is filled differently than envisioned above, this will impact the actual cost model and the price. Recall that there are three different roles, namely reuse engineer, certifier and reuser. The aim of the business model design is to allocate all tasks effectively over the components of the business model. The aim of the cost model is to organize these tasks so that they can be performed at minimal costs.

The more work that is actually being carried out by the reuser himself, the less work will be done by the

reuse engineer, and hence a lower price can be asked since the total effort related to the service provided is lower. The probability of the reuser having a more active role than in the stylized model is higher in the earlier phases of the RODE Process. In reality the reuser himself is likely to be quite involved in the process definition, which saves work for the reuse engineer and will be reflected in the price. On the one hand this decreases the costs for the Reuse engineer, on the other hand transaction costs increases and these should be taken into account (e.g. communications, test runs etc). The higher up one comes in the process, the more likely it is that the reuse engineer actually performs the larger fraction of the actual work, or that if the reuser is involved or consulted, that this increases the efforts in terms of communication, and that the effort estimates are accurate. Further, since the majority of effort is related to the later stages, the impact on price should be comparatively small.

6. RISK MANAGEMENT

The purpose of this section is to propose a generic risk management strategy encompassing the necessary mechanisms able to handle risks mainly attributed relating to the constant evolving nature of the Open source software paradigm as well as to the characteristics of involved external communities that provide inputs to the innovation process.

6.1 RISK MITIGATION

Selecting software components in a project influence the system quality attributes such as usability, security, operability etc. Selecting suitable software entities to adopt, based on the desired quality, can be a challenging task, as along with the desired characteristics, come risks. In the Open Source Software (OSS) realm, a vast number of available projects and components are available for reuse and incorporation into new software systems. However, this increased availability renders the selection decision problem. To alleviate this, various software evaluation methods have been proposed, (ie. BQR, SQO-OSS, OSMM, etc) offering assistance in measuring various qualities and aspects of software components. These approaches however, offer a time static (one point in time) quality evaluation without taking into account the living nature of open source development process, the community evolvement nor the implicit changes in quality (positive or negative) over multiple projects' releases.
In adopting the right components the reuse engineer should be able to examine the project's insights and make quick and sound judgments regarding project's maturity and overall acceptance/success. Foremost and for obvious reasons the reuse engineer would like to know the chances for the survivability of the project the candidate component belongs to, or in other words whether the project is doomed to failure.
Survival analysis can be effectively used as a generic risk mitigation approach to assess the duration and the survivability probability an open source software project. Research work in this direction has been proposed recently in [14] and [15], examining the application of duration analysis in software projects, while in [16], a statistical methodology based on a combination of fitted logistic regression and discrete event simulation for determining the "death" of a process based activity based on lack of progress, has been proposed. A recent update elaborating on survival analysis is the work proposed in [17], where the authors examined the lifetime of volunteer contributors and its impact on the continuity of an open source project.

In the Open-SME context we propose to use the survival analysis as presented by authors in [18]. The method mechanism is as follows:

1. Collection of data concerning the project the desired component resides in, from the project's CVS (concurrent versioning system). This data may include the following measurements:

Data collection measurements
Total number of commits
Max. commits per month
Min. commits per month
Avg. commits per month
Total number of actions
Max. actions per month
Min. actions per month
Avg. actions per month
Total number of committers
Total number of authors
Total number of unique files

2. Definition of the desired time slot for the statistical analysis defined as the time to the occurrence of a predefined, terminal event. This terminal event can be either project's "death" or failure. Duration time or simply duration is referred to the time from birth month

to the death month. With the term birth month we refer to the very first instance of activity in the project's CVS repository while death month refers to the month that the terminal event occurred. In our context the terminal event can be of two types:

1. The project is considered not active and thus it is abandoned.
2. The project has suddenly no code activity.

3. Calculation of the distribution of the duration described by three functions:
   1. Probability Density Function (PDF)
   2. Survival Function (SF)
   3. Hazard Function (HF)

The probability density function, (PDF) defined as the as the limit of the probability that a project is completed in the time interval (t, t + Dt) per unit width. The mathematical formula defining PDF is:

$$f(t) = \lim_{\Delta t \to 0} \frac{P(t < T < t + \Delta t)}{\Delta t}$$

The Survival Function (SF) is denoted by S(t) and is defined as the probability that the duration of a project is longer than t:

$$S(t) = P(T > t) = 1 - F(t)$$

S(t) is a non-increasing function of time t with the properties:

$$S(t) = \begin{cases} 1 & \text{for } t = 0 \\ 0 & \text{for } t = \infty \end{cases}$$

The Hazard Function (HF) is denoted by h(t) and is defined as the probability of completion during a very small time interval, and measures the tendency for project completion as a function of the duration of the project, assuming that the project is still active by the beginning of the interval:

$$h(t) = \frac{f(t)}{1 - F(t)} = \frac{f(t)}{S(t)}$$

Having the results from the formulas above the final step is to define the thresholds that will assist reuse engineer in making the optimal decision regarding the adoption of the component under study. Based on the outcomes the possible recommendations should be:

a) Adopt (as the project has high chances of survival in the given time frame – support will be provided)
b) Further Investigate (as the project's survival is questionable) – Sensitivity analysis and Simulation Techniques (i.e: Monte Carlo) should be applied, on data collected from project's releases.
c) Do not Adopt (as the tendency for project completion "death" or "fail" is high – no support is expected)

It is to be noted here that different thresholds should be applied in different application domains. These thresholds will be defined and periodically corrected, upon successful completions of the domain engineering process.
The benefits from applying this kind of analysis in projects of interest, is that will help reuse engineer to identify the overall ratio of persistence of community users of a given OSS project and therefore giving a simple and informative picture of the current hustle and bustle of users in this project. Additionally, will allow the establishment of precise expectations about the project's future evolution, under the assumption that the situation will not vary significantly from the current trend.

7. CONCLUSIONS

Creating the foundations for SME-AGs and SMEs to offer efficient tools and service for the easy and inexpensive reuse of OSS and OSS components is a challenging task, as complex processes in order to identify requirements and to develop the right tools must be developed and implemented, the socio-economic context conditions must be known, costs must be identified and quantified, and risks must be identified and mitigated. Other works in WP2 have shown that the required domain engineering process can be organized efficiently and create the tools needed in order to achieve the technical goals of OPEN-SME.

The socio-economic context conditions of SMEs and especially SME-AGs appear to demand a sound analysis of the position and capacities of SME-AGs within business ecosystems and value networks before business models for OSS reuse tools and services are developed. This might require a restructuring in order to change the role of some SME-AGs from a lobbying institution to a partner in value networks. In the case of the OPEN-SME business model, initial surveys have been carried out, and additional surveys are planned in the initial phase of the business model implementation in order to get a better overview and identify contact points for the initial markets.

An important success factor for the offering of tools and services for the reuse of OSS is that the vendors of these services – SME-AGs and SMEs – become able to assess the demand and the capacities of their potential clients. This requires detailed knowledge of business models and business strategies, both related to OSS. The foundation for assessing these characteristics has been provided in this paper.

Given the deep knowledge and personal trust to company and community members that plays a role for many SMEs when decisions about reuse of source code or software components are made, it appears necessary for a provider of reuse services to demonstrate that he is capable of understanding precisely what code and components are searched, for what purpose they are searched, and how they are going to be implemented, although the service provider might not belong to the company and maybe not even to the (inner circle of) the OSS community in or for which reusable code or components are wanted. In other words, the reuse service provider should be able to demonstrate that the specific knowledge assumed by the potential client to be necessary to conduct a proper reuse analysis on source code or software components is, in fact, not necessary, or at least not for all phases of the RODE process. This is the point where the core value of the OPEN-SME business model is located, as OSS reuse expertise, domain expertise, and the related engineering expertise can be provided together with a unique OSS reuse repository and a highly integrated toolset that offers so far unreached quality of OSS code reuse analysis.

Hardware vendors should not be overlooked when markets for OSS reuse tools and services are searched. They seem to have a demand for OSS reuse because they rely on software but do not have the deep knowledge of it and just want to see their hardware functioning.

There is a widespread uncertainty about the monetary value of OSS reuse services offered by a SME-AG, as even those companies that considered such offerings as a realistic alternative to their current reuse practices and were willing to pay for them were not able to quantify how much they would pay for these services. Together with the rather vague estimations of the time that should be dedicated to code and component analysis this observation suggests that many companies are not really aware of the costs that these activities generate. If this is the case, it is evident that the benefits of OSS reuse cannot be calculated properly, either.

The cost factors of tools and services for the reuse of OSS have been identified and methods to quantify these costs are provided in this document. The OPEN-SME business model

It could be shown that risks aligned with the development and deployment of the tools and services can be countered effectively.

Given all these opportunities and constraints, the OPEN-SME business model has been designed as a community-based model with a strong role of VSP as initiator and "enabler" during the implementation phase. Another key partner in the initial phase is AUTH because the AUTH team holds the technical and software engineering expertise that is necessary to make sensible use of the OPEN-SME repository and tools.

There is a clear-cut strategy for the implementation and testing of the OPEN-SME value propositions and the later roll-out of the business model, which will be performed in two directions, from local to national and international scope and from selected domains (robotics) into other OSS-reuse intensive domains. All SME-AGs and SME associations that participated in OPEN-SME play a role in the implementation and roll out of the OPEN-SME business model, some as distributors and multipliers, others as developers and / or service providers, and VSP in the role of the "CEO" of the OPEN-SME business.

REFERENCES

[1] Helander, N., Rissanen, T., "Value-Creating Networks Approach to Open Source Software Business Models", Frontiers of e-Business Research, 2005, 840-854.

[2] Moore, J. F., 1996. The Death of Competition: Leadership & Strategy in the Age of Business Ecosystems. New York: HarperBusiness.

[3] Normann, R., Ramirez, R., 1993. "From value chain to value constellation: designing interactive strategy." Harvard Business Review, Vol. 71 (1993), No. 4, 65-77.

[4] Mariotti, J. L., 2002. "The value network". Journal of Executive Excellence, Vol. 19, no. 7.

[5] Ghosh, R. A., 2006. Study on the economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU. Available online at: http://ec.europa.eu/enterprise/sectors/ict/files/2006-11-20-flossimpact_en.pdf

[6] Lakka, S., Stamati, T., Michalakelis, C., Martakos, D., 2011. "The Ontology of the OSS Business Model: An Exploratory Study." International Journal of Open Source Software and Processes, Vol.3, No. 1, 39-59.

[7] The 451 Group, 2008. Open Source is not a Business Model. New York et al.: 451 Group.

[8] Horsti, A., 2007. Essays on electronic business models and their evaluation. Helsinki: Helsinki School of Economics. Available online at: http://hsepubl.lib.hse.fi/pdf/diss/a296.pdf

[9] Osterwalder, A., 2004. The Business Model Ontology, A Propositional in a design science approach. Universite de Lausanne, Ecole des Hautes Etudes Commerciales. Available online at: http://www.hec.unil.ch/aosterwa/PhD/

[10] Al-Debei, M. M., and Avison, D., 2010. Developing a unified framework of the business model concept. European Journal of Information Systems, 19(3), 359-376.

[11] Amit, R. & Zott, C., 2001. Value creation in e-business. In: Strategic Management Journal, Vol. 22, No. 6-7, pp. 493-521.

[12] Teece, D. J., 2010. Business models, business strategy and innovation. Long Range Planning, 43, 172-194.

[13] Morris, M., Schindehutte, M., Allen, J., 2005. The entrepreneur's business model: Toward a unified perspective. Journal of Business Research, 58 (6), 726-735.

[14] Timmers, P., 1998. Business models for electronic commerce. In: Electronic Markets, Vol.8, No. 2, pp. 3-8.

[15] Osterwalder, A., Pigneur, Y., Tucci, C. L., 2005. Clarifying business models: origins, present, and future of the concept. Communications of the Association for Information Systems, 16, 1-25.

[16] Casadesus-Masanell, R., and Ricart, J. E., 2010. From strategy to business models and onto tactics. Long Range Planning, 43, 195-215.

[17] Zott, C., and Amit, R., 2010. Business model design: an activity system perspective. Long Range Planning, 43, 216-226.

[18] Chesbrough, H. & Rosenbloom, R. S., 2002. The role of the business model in capturing value from innovation: evidence from Xerox Corporation's technology spin-off companies. Industrial and Corporate Change, Volume 11, No. 3, pp. 529-555.

[19] Trombly, R., 2000. E-business models. In: Computerworld, Vol. 34, No. 49, pp. 61–62.

[20] Schlachter, E., 1995. Generating revenues from websites. In: Board Watch, July 1995. Available online at: http://boardwatch.internet.com/mag/95/jul/bwm39.html.

[21] Rappa, M., 2004. Business Models on the Web. Available online at: http://www.startupjunkies.org/business_models.pdf

[22] Linder, J C., and Cantrell, S., 2000. Changing business models: surveying the landscape. Accenture Institute for Strategic Change.

[23] Margretta, J., 2002. Why Business Models Matter. Harvard Business Review, 80 (5), 86-92.

[24] Osterwalder, A. & Pigneur, Y., 2010. Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers, Wiley.

[25] Ho, Y. C., Fang, H. C and Hsieh, M. J., 2011. The relationship between business model innovation and firm value: a dynamic perspective. World Academy of Science, Engineering and Technology, 77, 656-664.

[26] Chesbrough, H., 2006. Open business models: How to thrive in the new innovation landscape. Boston, MA: Harvard Business School Press.

[27] Chesbrough, H., 2010. Business model innovation: opportunities and barriers. Long Range Planning, 43, 354-363.

[28] Kudorfer, F., Laisne, J. P., Lauriere, S., Lichtenthäler, J., Lopez, G., and Pezuela, C., 2007. 'State of the art concerning business models for systems comprising open source software'. Available online at: www.qualipso.org/media/A2/D2.1.3.pdf

[29] Daffara, C., 2008: Business models in FLOSS-based companies. Available online at: http://ifipwg213.org/system/files/OSSEMP07-daffara.pdf

[30] Dornan, A., 2008. The Five Open Source Business Models. Available online at: http://www.informationweek.com/blog/main/archives/2008/01/the_five_open_s.html

[31] Drucker, P. F., 1994. Theory of the Business. Harvard Business Review, Vol. 7, 95-104.

[32] Osterwalder, A. and Pigneur, Y., 2002. "An eBusiness Model Ontology for Modeling eBusiness". BLED 2002 Proceedings. Paper 2.

[33] Glott, R., Haaland, K., Ghosh, R. A., Deprez, J. C., 2010. Validation of Data and Measurements of Advanced F/OSS Projects. QualOSS Project Report. Available online at: http://www.qualoss.org/deliverables/WP3-D3.3_final_submitted.pdf

[34] van Gurp, J., Prehofer, C., and Bosch, J. 2010. Comparing Practices for Reuse in Integration-oriented Software Product Lines and Large Open Source Software Projects. In: Software - Practice and Experience, Vol. 40, no. 4, pp. 285-312.

[35] Brown, A., and Booch, G., 2002, Reusing Open-Source Software and Practices: The Impact of Open-Source on Commercial Vendors. In: Software Reuse: Methods, Techniques, and Tools. Lecture Notes in Computer Science, Volume 2319/2002, pp. 381-428

[36] Mockus, A. 2007, Large-scale code reuse in open source Software, FLOSS '07 Proceedings of the First International Workshop on Emerging Trends in FLOSS Research and Development, IEEE Computer Society Washington, DC, USA. Available at: http://mockus.org/papers/ossreuse.pdf

[37] Pedersen, P. 2006, Best Practices for Managing Code Reuse, Black Duck Software, available at: http://www.blackducksoftware.com/files/webmedia/_Slides/Code-Reuse-Slides.pdf

[38] Chang, V., Mills, H., Newhouse, S., 2007 "From Open Source to long-term sustainability: Review of Business Models and Case studies". All Hands Meeting 2007, OMII-UK Workshop, 10 September -

13 September, 2007, Nottigham, UK. Available online at: http://eprints.ecs.soton.ac.uk/13925/1/ICVC_HRM_SN_AHM_final_paper1b.pdf

[39] Porter, M. E., 1996. What is Strategy. Harvard Business Review, November-December 1996, pp. 61-78.

[40] Kano, N., Seraku, N., Takashi, F., Tsuji, S., 1984. Attractive Quality and Must-be Quality. The Journal of the Japanese Society for Quality Control, Vol. 14, No. 2, pp. 39 -48., cited from: Sauerwein, E., Bailom, F., Matzler, K., Hinterhuber, H. H., 1996: The Kano Model: How To Delight Your Customers. Preprints Volume I of the IX. International Working Seminar on Production Economics, Innsbruck/Igls/Austria, February 19-23, 1996, pp. 313 -327. Available online at: http://faculty.kfupm.edu.sa/CEM/bushait/CEM_515-082/kano/kano-model2.pdf

[41] Mohan, M., 2011. Marketing to Early Adopters. Available online at http://www.pr-squared.com/index.php/2011/08/marketing-to-early-adopters

