Electronic Communications of the EASST
Volume 48 (2011)

Proceedings of the
Fifth International Workshop on
Foundations and Techniques for
Open source Software Certification
(OpernCert 2011)

Process Scenarios in Open Source Software Certification

Fabrizio Fabbrini, Mario Fusani and Eda Marchetti

15 pages

Guest Editors: Luis Soares Barbosa, Dimitrios Settas
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Process Scenarios in Open Source Software Certification

Fabrizio Fabbrini[1], **Mario Fusani**[1] **and Eda Marchetti** [2]

[1] (fabrizio.fabbrini, mario.fusani)@isti.cnr.it,
Systems and Software Evaluation Centre, ISTI - CNR Pisa Italy
[2] eda.marchetti@isti.cnr.it, Software Engineering Laboratory, ISTI - CNR Pisa Italy

Abstract: Certification of Open Source Software (OSS) presents inherent trade-offs due to the necessity of precisely identifying both a product and an independent certification agent, and on the other of maintain the peculiar, valuable OSS characteristic of being available to an unlimited multiplicity of actors for trial, use and change. This is an intriguing challenge, usually solved by removing from the picture the certifying agent and providing an intrinsic certification by means of rigorous, re-applicable property demonstrations, adopting Formal Methods (FM) in expressing and verifying the code. As such approach, yet quite valuable and good-promising, has some restrictions (such as the limited set of provable product qualities), we propose to tackle the problem by analysing the various processes executed by different OSS stakeholders, including the process of an independent Certification Body. In the paper some kinds of representative scenarios in which such processes interleave are presented and discussed. The aim is to introduce a process-centered perspective for OSS that can stimulate research to further understand and mitigate the mentioned trade-offs.

Keywords: Open Source Software, Certification, Software Process

1 Introduction

Traditionally, software certification has been mainly associated with proprietary software or Closed Source Software (CSS) with the aim of increasing the confidence that a software-related product or service actually possesses its declared behavioral and/or structural attributes. Recently, the increasing adoption of Open Source Software (OSS) in new environments, such as public administrations, makes it even more urgent to evaluate the correctness and other software quality attributes, such as reliability and usability, of the software used. Indeed, intrinsic product variability, context criticality, compliance to standards and typical constraints of specific domains evidence that certification is still a key factor in adopting OSS software.

Commonly available solutions to this problem try to remove the activity of an independent certifying agent and provide intrinsic certification by means of rigorous demonstrations of software properties by adopting Formal Methods (FM) in expressing such properties and verifying the code against them. However as pointed out by [Wal04], one task is to certify properties of a software item with respect to defined specifications, and another task is to certify related properties of a system (hardware and software) of which the software item is a continuously evolving component. Sometimes the whole system is not available, and also when it is, it is not always easy just to express the ”global” properties of a software component (typically detectable as their impact into external system qualities), let alone to verify them.

Out of the FM, Model Checking techniques are typically adopted to demonstrate the characteristic of *correctness* (with sub-characteristics expressed in the form of provable/non-disprovable properties such as *liveness* and *safety*). Then Model Checking, now rather feasible and intensively automated, can be used for certification [Wal04] (an overview of some proposals is in Section 2). However these approaches, rigorous and good-promising (thanks to availability of many tools)as they may be, cannot be extensively adopted because the set of provable product qualities is still limited and depends on the available specifications. Moreover a recent survey [HSI10] on ongoing OSS projects shows that testing is still almost the only way adopted to check product properties, even if these properties are validated only for the test conditions.

From this considerations our proposal wants to tackle the problem by analyzing the various processes executed by different OSS stakeholders, including an independent Certification Body.

Starting from the assumption that OSS products have interesting, although unconventional, common process features that can be taken into account for certification, we introduce in this paper an approach to a sort of formalized view of the OSS certification activities. Motivated by recent trends in the Italian reality of the Public Administration, that fosters the use of OSS in automation of public offices, we want to investigate different related scenarios, evidencing the roles of the playing actors to find possibly standardizable yet feasible conditions that make an OSS eligible for certification.

We want to focus the reader's attention on the processes performed during the OSS life-cycle to stimulate research towards further understanding and mitigation of the trade-offs between the typical discipline requested by certification and the unconstrained nature of OSS that makes it so appealing and objectively valuable. In this process-centered perspective, we show how the long-dated certification concept, gained with conventional products, can be re-defined and applied to the more complex and varying OSS scenarios. We therefore highlight some of the OSS development process evidences that can be used in a certification process and show that, even inside different scenarios, these processes do have interesting, although unconventional, common features that can be taken into account for product certification.

The paper is organized as follows: In Section 2, a sample of related literature is commented in the light of our objectives and the typical characteristics of the OSS are summarised. In Section 3, the concept of certification is re-visited and new evidences are proposed. In Section 4, significant OSS certification scenarios are presented and commented to point out the role of the main actors, their expected actions and mutual relationships. In Section 5 an analysis of the proposed scenario is provided while Conclusions are drawn in Section 6.

2 Related work

From the vast literature of OSS and certification, we selected some works describing the current trends. As noticed, OSS certification is often related to the use of FM [KM08] in transforming requirements into code and code into requirements, as it is the case of safety and security related properties [CS08, SC09]. Model Checking is a further proposed solution, sometimes considered as the most effective technique for OSS analysis [CGR09]). Alternative proposals are those focused on agile methodologies for keeping consistency between a product and the evidences of the product(for instance [CGR09, MTT09]).

Even though effective, the so far mentioned solutions are limited to specific properties of the OSS software and can be dependent on the specifications.

The advantages of independent certification management is introduced in [PB08], which suggests also mechanisms for doing it (granting/revoking certificates and performing continuous certification in a vulnerable environment) and enforced in [KKS10], where an independent body (in the case, an association of developers and/or users) is expected to provide on-line services to OSS component integrators for which a set of tools, also including reverse engineering tools, search engines and analysers, are being produced.

However, so far the proposals for independent certification have been rather vague and certification against specific standard is reported as a drawback. In our view, certification can advantageously be performed at various levels and an independent entity, the Certification Body, can play different roles in its peculiar task of confidence and transfer it among stakeholders. The scenarios we propose try to figure out the various aspects and roles involved in a certification process and address issues for further research.

3 Peculiarities of the certification of OSS

The literature analysis and the experience in product (CSS) evaluation and process (SPICE) assessment of our Centre [ISO08b] help us to highlight some peculiarities of the OSS that should be taken into consideration for an OSS certification process, typically: availability of usually free or not expensive source; possibility of downloading the OSS from a public website; use of a development environment managed by a community of developers/testers/users, that eases rapid code change and re-use; possibility of measuring product characteristics such as correctness, reliability and maintainability.

From this picture, the main factors influencing a OSS certification process can be summarized as:

- **Stakeholders**: users and developers happen to work closely together and the boundaries among the roles become much more indistinct.
- **Requirements specifications**: In OSS, specifications are not any more controlled by a single organisation and continuously evolve according to the needs of individuals or companies. Often they can be collected from various sources such as developers forums or test cases.
- **Verification and testing**: OSS properties mostly get verified by testing in operational environment, both in case of software component selection, and during actual service. Testing before delivery is then only a fraction of the testing process. Regarding static verification, we noticed in Section 1, about FM techniques, the phenomenon of many proposals in literature and scarcely adopted in practice. Moreover, verification of process documents, which in CSS is one of the best sources of information for a Certification Body, is only marginal in OSS.
- **Independent development**: OSS can be totally or partially cloned by various developers, also concurrently. This may rise new configuration management problems.

- **Traceability of products**: OSS are characterized by high variability and evolution (versioning) of the same product, evidencing some difficulties in identifying the product from its releases.

- **Configuration management**: configuration management life-cycle processes are still there ([OMK08] and [HSI10]), and their actors are generally geographically distributed.

- **Process-related work products**: these are documents such as FAQs, annotations, lessons learnt, bug logs, and so on, which evidence the importance and the interest for OSS inside the community and the development effort behind the completion of the OSS itself. A Certification Body should learn to deal with these work products instead of the traditional CSS documentation.

From the above characteristics it becomes clear that the certification process is expected to continuously track evolving OSS requirements specifications and take evidences from the operational environment. Operational testing [L+96], that in CSS is extremely difficult to manage for cost and time constraints, becomes common practice ([MTT09], [HSI10]) and a precious contribution for improving the overall confidence in the OSS product.

Thus one of the most important role in the assessment part of certification is played by the process-related work products such as blogs, FAQs, annotations, bug logs and so on. That source of information can allow monitoring of product maturity, testing activity and properties implementations and is an essential element of the certification process. In particular, such information aimed to improve or decrease confidence about the product properties can be discovered by verification techniques such as dynamic testing, static analysis and Model Checking. To perform this process-related assessment, traditional document analysis cannot be used and new techniques must be devised, not excluding mining and natural language processing of raw text, as, for example, an evolution of the research proposed in [YSJS07].

Certification process could also be influenced by independent development: versioning of the same OSS and the different abilities of different developers are factors that can affect the final decision of a Certification Body.

These evidences, as well as any other pieces of information derivable by the current available OSI certification [OSI08] have to be monitored in a certification process. In particular, due to the dynamic nature of OSS products, the natural consequence is that certificates are associated to a certain evolution of the OSS, thus related to a specific time and version.

To collect the certified versions of the same OSS products, while the certification is valid, and to reduce the complexity of the communications among the stakeholders, the concept of a *virtual repository* can be introduced. Interactions with a virtual repository can happen only on a voluntary basis for example to get a (possibly dynamic) certificate. This way, registering an OSS project to a virtual certification repository would be appealing and would ease the adoption of the software itself.

The analysis of the influencing factors evidences that independent certification need new standards that do not disrupt the characteristic of free development while allowing an OSS to be eligible for certification. These new standards should recognize collaborative working environments and propose rules to properly collect process evidence and certified products.

In Table Table 1 some differences and similarities between CSS and OSS certification processes are shown. In this partial list, the only significant similarity seems to be CB accreditation.

4 The scenarios context

Lifecycle aspects for OSS certification have been extensively described in literature [Tay09].

In this section we provide an attempt at schematizing two different certification scenarios considering, as mentioned in the Introduction, the context of the Italian Public Administration (PA) environment. As we cannot validate yet our proposal, representing and analyzing possible certification scenarios seems useful to see how the consequences of the factors mentioned in Section 3 can work together. Thus even if specific, we think that these domestic situations, once analyzed, can have many points in common with other international environments and can be easily generalized or adapted to any other context.

In Italy, as in other countries, there is an increasing interest by the central Government in adopting OSS systems for the development of public projects, in particular in the Public Administration context, so to keep costs reasonable and accelerate the completion of administrative system projects. However, due to some criticalities of the context and the many standards and constraints specific to the public environment, only certified products are eligible to be integrated or used in the existing systems. Thus certification is still recognized as a key factor to persuade the various PA offices to adopt OSS solutions. This opens up two possible scenarios:

In the first one the PA itself, to assure that OSS is compliant with the standards and the required level of quality, commissions the certification to an external Certification Body (CB)).

In the second one, it is thinkable that developers can make available ready-to-use, certified OSS to the PA, under the payment of a (perhaps only symbolic) certification fee. In this case even if the developers have to face certification expenses from the CB, they can have incomes from the widespread use of the certified product, as well as from installation and maintenance activity.

We schematize the first scenario in section 4.1 where we consider the PA as Customer/User stakeholder and the certification ordered to an external Certification Body,

The second scenario is presented in section 4.2 where OSS Developer(s) would like to take the advantage pushing in the adoption of the OSS from the PA, by proposing OSS certified products on the market.

4.1 Certification scenario: PA triggering the certification process

In this section we provide details about a possible scenario in which the PA promotes the certification of an OSS product that needs to be integrated in its administrative systems. In particular in Figure 1 we schematize the main stakeholders of the scenario to outline their mutual relationships.

We suppose the PA uses a public OSS repository where developers provide OSS (OSS source artefacts in the figure) implementing different functionalities. The Certification Body (CB) has the role to certify, according to PA requests, conformity to applicable standards and quality constraints involved with the selected OSS and to maintain a common certified OSS repository. With

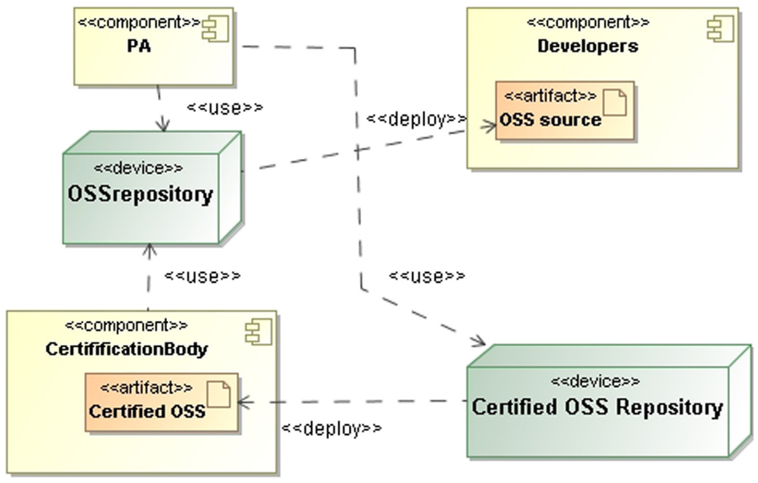

Figure 1: Simplified stakeholders scenario with PA

no expectations to be exhaustive, a possible interaction between the above mentioned stakeholders during the certification process can be schematized as in Figure 2.

Developers and PA, from different points of view, contribute to the requirement elicitation activity, which basically produces a list of constraints that can vary from operational systems specification to development environment constrains, performance and quality attributes and other more specific requests about the functionalities to be implemented. Following the OSS philosophy we consider not restrictive to represent the requirements elicitation as a free, open and not ruled activity, where exigencies coming from different actors and environments join together in common, publicly available (possibly textual) documents.

The various requirements can then be implemented in parallel into OSS product(s) or updated and refined by the PAs. As a common practice, the OSS repository will contain for each OSS product the source code and possibly the OSS storyboard, i.e. all the available source of information concerning the released OSS (logs, comments, description of functionalities and so on). PA can therefore select from the OSS repository the OSS product considered eligible for certification and commission to CB the analyses and the management necessary for the certification itself. CB, following the standard procedure, will perform requirement analysis, possibly recovering data defined during the requirements elicitation, and collect all the available OSS information (as described in section 3). Then the CB can continue the certification activities that, as said in Section3, may require further interaction with the Developers and with independent verification laboratories. The process, whose requirements are listed in Section 5, can produce three principal possible results:

1. CB declares the OSS product not eligible for certification;
2. the certification is successfully concluded, with the certified OSS transferred into the COSS repository and the storyboard opportunely updated;
3. CB identifies required modifications or bugs fixing needs and communicates this to PA

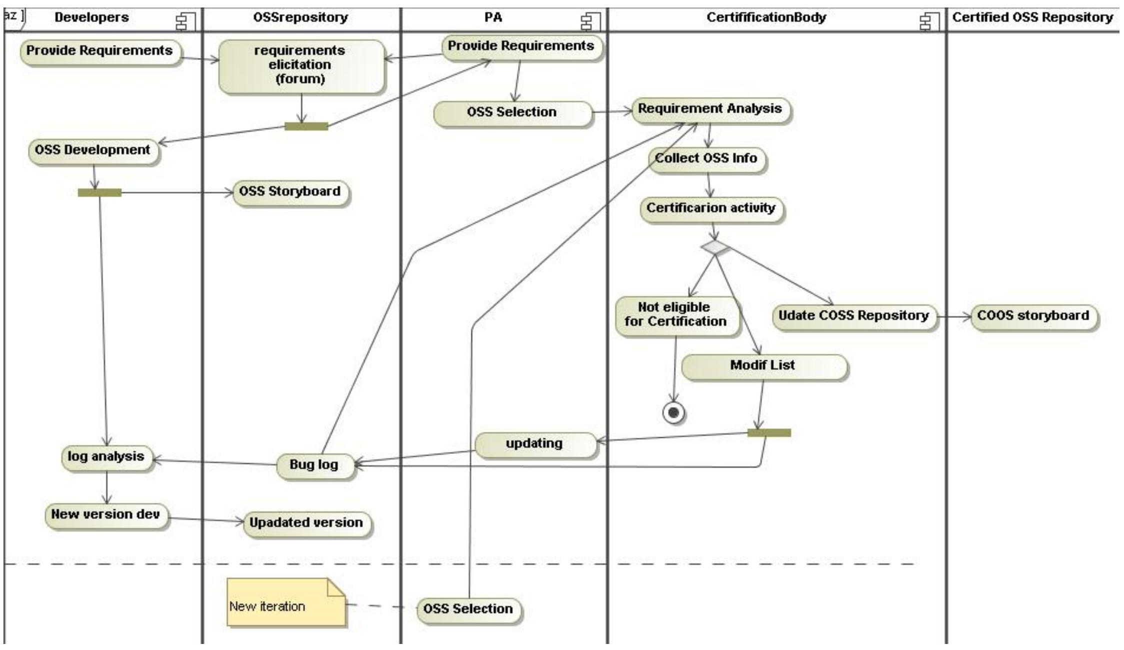

Figure 2: Simplified interaction scenario with PA

that has commissioned the certification. This in turn can update the OSS storyboard and possibly the bug log of the OSS repository so that developers can release an improved version of the required system and let the certification restart for a new iteration

Of course there are variants, still in the perspective of continuous certification ([PB08], [CGR09]): for example, in case of successfully concluded certification, Developers may go on independently updating the software, whose evolved version could draw the interest of the same PA or another public / private institution, which would trigger again the certification process.

From this simple scenario the following considerations can be drawn:

- The stakeholders act rather independently (see Section 3), even if they must synchronize on various occasions. The repository itself, a passive entity, acts as a common channel.
- CB, differently from all the other stakeholders, is bound to behavioral rules. As we already pointed out, it could (and should) be compliant to severe, even conventional standards, but this does not make its presence in an OSS scenario disturbing.

This scenario is a rather high-level one, and its representation hides many intermediate steps that would make the complete description quite a job (Requirements elicitation, Requirements analysis, Certification activity and so on) that can be omitted here.

4.2 Certification scenario: Developer triggering the certification process

In this section we provide details about a scenario in which the promoter for the certification of an OSS product is the Developer. Here the main role of PA is to express the user needs and interest in terms of requirements, while the role of Developers is to implement certifiable products. Certification of OSS against the many standards and constraints specific to the public environment is considered a key factor to convince the various PA offices to adopt already developed OSS. As a side effect, the ability to provide accredited products for the PA could be a means for the developers to increase the number of clients and to make profits (for instance from software installation, maintenance or other related activities).

The stakeholders in this scenario are the same as in the previous one (Figure 1). Thus, again PA uses the public OSS repository for requirements elicitation and Developers are in charge of the OSS implementation (OSS source artefacts in the figure).

On behalf of the Developer, and not of PA as in the previous scenario, CB certifies the OSS according to PA standards and quality constraints, and maintains a common certified OSS repository. It is then possible to suppose that PA can download the certified OSS software from the certified repository upon payment of a special symbolic fee.

We schematize the interactions between Developers, PA and CB during the certification process as shown in Figure 3.

As in the previous scenario, PA contributes to the requirement elicitation activity, which mainly consists of a list of constraints, quality attributes and/or other more specific requests about the functionalities to be implemented. Then the various requirements can be implemented in parallel into OSS products for which the source code and possibly the OSS storyboard, similarly to the previous scenario, are made publicly available. At this step in the process only the developers who want to certify a version of OSS developed product order the certification analysis to the Certification Body.

Note that in this case, CB can use possible information about the development process provided by the Developer itself. Thus, factors that are hardly exploitable in OSS certification, such as, for instance, reference standards, life-cycle traditional standards, architectural requirements, or internal quality characteristics can now be used by CB for certification. As a consequence, the CB activity could be closer to that of a traditional CSS certification process.

Thus, CB, following the standard procedure, will perform a requirement analysis, integrating data about requirements elicitation (if any), available OSS information and additional information provided by Developer. Then CB can continue the certification process, as described in the previous scenario, directly interacting with Developer and possibly with independent verification laboratories.

Just as before, CB activity can produce three principal possible results: 1) CB declares the OSS product not eligible for certification; 2) the certification is successfully concluded, with the certified OSS transferred into the COSS repository and the storyboard opportunely updated; 3) CB identifies required modifications or bugs fixing and communicates this to the developer that has commissioned the certification. Only when the improved version of the required system is developed the certification process will restart for a new iteration.

Once committed in the certified OSS repository, the certified version can be used by Developer for its own marketing.

The stakeholders of this scenario act more independently than in the previous one, so minimizing the points of synchronization and better reflecting the typical features of OSS development. The OSS repository is still independently updated upon request of any developers who ask for certification. For exploitation purposes and also for covering the certification expenses, it is

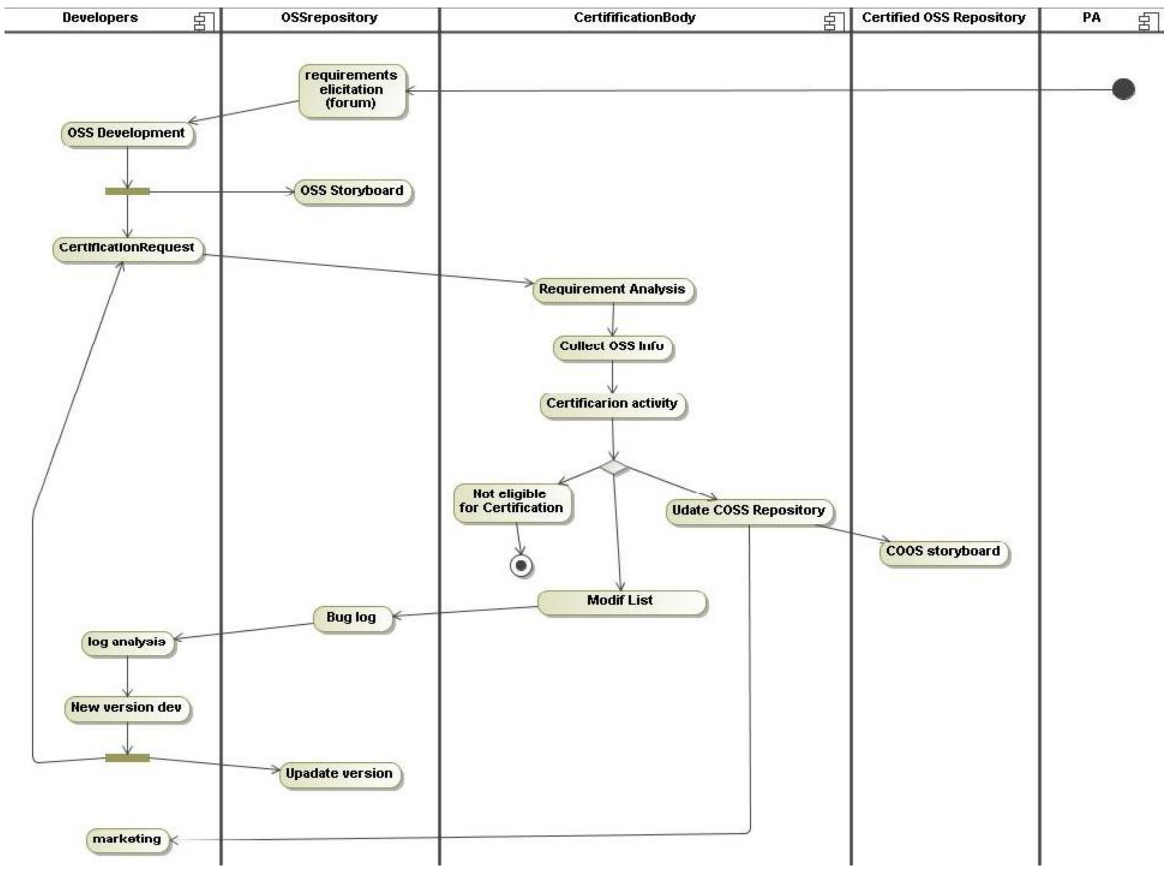

Figure 3: Simplified interaction scenario with Developer

plausible that developers ask each user of the Certified repository a symbolic fee.

The scenarios have been presented to a rather high abstraction level, purposely avoiding intermediate steps that would disturb viewing of the overall picture of the proposal.

5 Comparison and discussion

By describing in this paper only two typical OSS certification scenarios we intended to provide a first tentative to formalize the necessary steps of a the certification process, considering respectively two different triggering situations: one from the PA side and the other from the Developer's. Even if, in the considered scenarios, the overall process shows no development but service only, even a slightly deeper inspection can reveal that process can be composed of various, typically interacting, processes, whose characteristics can be inspired to a process reference model such as in [ISO08a]. In particular: the CB process structure includes lifecycle processes such as management (common to all stakeholders), requirements analysis, verification and testing; Developer process include a more complete set of lifecycle processes, among which coding; PA process may include requirements elicitation (present also in the other stakeholders), product acquisition and supplier monitoring. All these processes synchronize with each other, also through the OSS repository, a passive entity that must have its access rules. It is a very high level synchronization with rather weak coupling features, even less compelling in the second scenario, so no deadlock conditions may occur. So, the overall certification process can be expected to be feasible and repeatable, provided it can rely on consolidate best practices. Moreover, it has to be tailorable and able to be monitored even in dynamically evolving situations. Indeed, due to the intrinsic characteristics of OSS development, certification has to face the typical cactus-like versioning of the same product. As a a consequence, an important requirement emerging from both scenarios is that the certification policy should clearly establish the properties to be certified and the validity limits of the certificate itself.

From both scenarios, the need for of an independent Certification Body, able to assess the quality level and the standard compliance of the source code and possibly of other work products with respect to the PA constraints, emerged clearly. As already noticed, the CB is the only stakeholder that has to comply to strict behavioral rules, such as requirements expressed in [ISO04]. The main difference in the situations considered, is represented by the information that CB can use for its activities: In the first scenario (Section 4.1) available data are minimal and mainly represented by the process-related work products; In the second one (Section 4.2) CB could exploit also evidences about the development process provided by the developer itself.

Summarizing, CB should be responsible for the following technical activities, that it might also directly/indirectly perform.

- Assessment and verification of properties declared in a certification scope. Specifically for the scenario presented in Section 4.1, when little or no reference-model exists, such as quality / performance / functional requirements or expected attributes, the role of CB is more exploration than verification: in this case no certificate may be issued, as is not in case the verifications fail (see Figures 2).

- Witnessed or monitored testing. In particular for the first scenario testing is executed by actors different from CB, such as Developers or Users. If testing procedures and reports are standardized and automated, then the monitoring part takes less effort by the CB. This could be true also for the second scenario (Section 4.2), but the strict collaboration from CB and developer could assure more suitable testing info and make the certification process easier.

- Independent testing (e.g., executed by an accredited Independent Laboratory, possibly conformant to ISO/IEC 17025. This is possible in both the scenarios considered.

- Code analysis and inspection (possibly executed by reviewers independent of Developers)

- Model Checking (based on formal models as a source) and Software Model Checking (based on code as a source).

Regarding testing, we already observed that in OSS most of this activity is focused on operational testing (Section 3), that typically can be considered as software validation.

Finally the two scenarios are different also for the purposes of the certification process. In the first scenario (Secton 4.1) certification is a guarantee for the PA that the standards and constraints specific to the public environment are not compromised or invalidated by the adoption of a OSS software. In the second scenario developers can exploit their certified ability in implementing accredited products for the PA for several beneficial side effects:

- Developers could increase the possible clients because their certified products can be adopted as a ready-to-use and cheap solution in the numerous PA(s) having the same necessities and constraints.
- The certified skill of the developers could be a precious advertisment within the PA communities, and in general for any other OSS community, to increase possible orders for different product development.
- Developers could ask for a symbolic fee for each download from the OSS repository of the Certified OSS. This could refund the developer of the expenses sustained for the certification.
- From the adoption of the OSS certified product, developers could exploit the possibility of opening a new business market due to product diffusion, installation and maintenance activities.

6 Conclusions

In this paper we examined the frequently discussed issue of Open Source Software (OSS) certification. As our Centre has been active in software product verification and process independent assessment since mid 1980's (both as an applied research activity and a service provided to Public Administration and privates), but never in OSS environment, we believe that this perspective could extend the Centre's scope of activity towards a promising business environment.

From the experience gained so far with the Centre, we think we know something about the basic nature of certification for traditional Closed Source (CSS) software, and we expect that the certification concept should be, in some parts, revised to adapt it to the nature of OSS.

In this paper we wanted to observe the process aspects of OSS certification, because of our experience in process engineering and because the process concept includes more knowledge and practice besides the set of techniques adopted.

Generally speaking, a process uses its resources, including technology in the aspects of: appropriateness for the purpose, ability of modeling real situations, ability of providing methods of operation, partial/full automation of such methods by means of appropriated tools (to be integrated into the process), related human factors (role specialization, knowledge, training, skill, motivation, again to be integrated into the same process) and ability of successfully deploying the technology in real projects. Out of these aspects, we addressed here some issues regarding key-roles behavior.

We first re-discussed some aspects of the certification in the light of OSS in terms of reference, standards, techniques, practices, role of stakeholders, examining what factors can positively or negatively impact on the main certification goals (Section 3), then projected the overall certification process into the single processes executed by some significant stakeholders (Developer, Certification Body and Customer in the particular case of Public Administration) (Section 4).

Table 1: Comparison of CSS and OSS certification process

Certification process properties, practices and techniques	**CSS certification process**	**OSS certification process**
Certification process compliant to standards (like ISO/IEC 17000)	Achieved through formal CB accreditation	Achieved through formal CB accreditation
Analysis and verification of life-cycle process documents, different from source code, such as: Requirements, architecture, verification, testing, design reviews,	Quite feasible and opportune for certification	Usually not available
Analysis and verification of life-cycle process documents different from source code such as: field information, forums, blogs,	Usually not provided and if so, unimportant with respect to traditional lifecycle work products	Mostly available and useful. Needs of some general rules and standards to
Analysis and verification of source code	Typically indirect: an assessment of developers' code verification work	Direct code verification using inspection tools and reverse engineering
Close relationship with developers	Feasible and useful	Quite loose relationships available, if any: virtual repository can be an indirect relationship
Continuous certification (multiple versions)	Difficult and expensive	Often necessary
Use of virtual public repository	Often impossible	Quite opportune

We made these processes, together with a passive process ”OSS repository”, interact in a couple of possible significant scenarios, representing them as activity diagrams, and analyzed some pros and cons of the scenarios, trying to summarize a set of requirements for the ”OSS certification process”.

We also pointed out how the process executed by an independent Certification Body, bound by accreditation duty to follow some rigorous standards, can help to mitigate the trade-off between the intrinsic properties of the OSS and the strict rules imposed by certification.

We think this can be another, perhaps untried-before way to have a higher-level view of the possible activities playing around OSS certification, that might be useful for search for more OSS process certification features.

We do not claim to be exhaustive in this formulation, nor particularly innovative. The aim of this paper is to introduce a process-centered perspective for OSS that can help to understand possible different scenarios and to stimulate related research issues.

Bibliography

[BCF+10] I. Biscoglio, A. Coco, M. Fusani, S. Gnesi, G. Trentanni. An Approach to Ambiguity Analysis in Safety-related Standards. In *Proc. of QUATIC 2010 (7th International Conference on the Quality of Information and Communications Technology)*. September 2010.

[Bou10] A. Boulanger. Open-source versus proprietary software: Is one more reliable and secure than the other? *IBM Systems Journal* 44(2):239–248, 2010.

[CGR09] C. Comar, F. Gasperoni, J. Ruiz. OPEN-DO: AN OPEN-SOURCE INITIATIVE FOR THE DEVELOPMENT OF SAFETY-CRITICAL SOFTWARE. 2009.

[CS08] A. Cerone, S. A. Shaikh. Incorporating Formal Methods in the Open Source Software Development Process. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. September 2008.

[FFL06] F. Fabbrini, M. Fusani, G. Lami. Basic Concepts of Software Certification. In *Proc. of 1st International Workshop on Software Certification (CERTSOFT'06)*. Pp. 4–16. McMaster University, 2006.

[Fus09] M. Fusani. Examining Software Engineering Requirements in Safety-Related Standards. In *Proc. of DeSSerT (Dependable Systems, Services and Technologies)*. April, 2009.

[HSI10] Z. Hashmi, S. Shaikh, N. Ikram. Methodologies and Tools for OSS: Current State of the Practice. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. September 2010.

[ISO96] ISO/IEC. ISO/IEC Guide 2:1996, Standardization and related activities General vocabulary. ISO/IEC, 1996.

[ISO04] ISO/IEC. ISO/IEC 17000: 2004, ISO/IEC 17000:2004, Conformity assessment - Vocabulary and general principles. ISO/IEC, 2004.

[ISO08a] ISO/IEC. ISO/IEC 12207 - Information Technology: Software life cycle processes. ISO/IEC, 2008.

[ISO08b] ISO/IEC. ISO/IEC 15504-5:2006 – Information technology – Process Assessment – Part 5: An exemplar Process Assessment Model. ISO/IEC, 2008.

[ISO08c] ISO/IEC. ISO/IEC TR 15504-6:2008 Information technology – Process assessment – Part 6: An exemplar system life cycle process assessment model. ISO/IEC, 2008.

[KKS10] G. G. Kakarontzas, P. Katsaros, I. Stamelos. Component Certification as a Prerequisite forWidespread OSS Reuse. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. September 2010.

[KM08] A. Khoroshilov, V. Mutilin. Formal Methods for Open Source Components Certification. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. September 2008.

[L$^+$96] M. R. Lyu et al. Handbook of software reliability engineering. McGraw-Hill New York et al., 1996.

[MTT09] S. Morasca, D. Taibi, D. Tosi. Towards certifying the testing process of Open-Source Software: New challenges or old methodologies?. In *Proc. of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development*. Pp. 25–30. IEEE Computer Society, 2009.

[OM09a] A. Ocampo, J. Muench. Rationale modeling for software process evolution. *Software Process. Improvement and Practice* 14(2):85–105, 2009.

[OM09b] A. Ocampo, J. Muench. *Rationale modeling for software process evolution*. Volume 14(2). 2009.

[OMK08] T. Otte, R. Moreton, H. D. Knoell. Applied Quality Assurance Methods under the Open Source Development Model. In *Proc. of the 32nd Annual IEEE International Computer Software and Applications Conference*. Pp. 1247–1252. COMPSAC, 2008.

[OSI08] OSI. OSI Certified Open Source Software. OpenSource.org, 2008.

[PB08] S. Pickin, P. T. Breuer. Open Source Certification. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. September 2008.

[SC09] S. A. Shaikh, A. Cerone. Towards a metric for Open Source Software Quality. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. September 2009.

[Tay09] R. Taylor. Understanding how OSS Development Models can influence assessment methods. In *Proc. of the Third International Workshop on Foundations and Techniques for Open Source Software Certification*. March 2009.

[Tri02] L. Tripp. Software Certification Debate: Benefits of Certification. *IEEE Computer*, pp. 31–33, June 2002.

[Uni] Unified Modeling Language. http://www.uml.org/.

[Voa00] J. Voas. Developing a Usage-Based Software Certification Process. *IEEE Computer* 33:32–37, 2000.

[Wal03] K. C. Wallnau. Volume III: A Technology for Predictable Assembly from Certifiable Components. Technical report, SEI - CMU, April 2003.

[Wal04] K. C. Wallnau. Software Component Certification: 10 Useful Distinctions. Technical report, SEI - CMU, September 2004.

[YSJS07] K. Youngjoong, P. Sooyong, S. Jungyun, C. Soonhwang. Using classification techniques for informal requirements in the requirements analysis-supporting system. *Inf. Softw. Technol.* 49:1128–1140, November 2007.