The RISCOSS Platform for Risk Management in Open Source Software Adoption

X. Franch[1](✉), R. Kenett[2], F. Mancinelli[3], A. Susi[4], D. Ameller[1], M.C. Annosi[5], R. Ben-Jacob[2], Y. Blumenfeld[2], O.H. Franco[1], D. Gross[4], L. Lopez[1], M. Morandini[4], M. Oriol[1], and A. Siena[4]

[1] Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{franch,dameller,llopez,ohernan,moriol}@essi.upc.edu
[2] KPA, Raanana, Israel
{ron,ronb,yehudablu}@kpa-group.com
[3] XWiki, Paris, France
fabio.mancinelli@xwiki.com
[4] Fondazione Bruno Kessler (FBK), Trento, Italy
{susi,gross,morandini,siena}@fbk.eu
[5] TEI - Ericsson, Rome, Italy
mariacarmela.annosi@ericsson.com

Abstract. Managing risks related to OSS adoption is a must for organizations that need to smoothly integrate OSS-related practices in their development processes. Adequate tool support may pave the road to effective risk management and ensure the sustainability of such activity. In this paper, we present the RISCOSS platform for managing risks in OSS adoption. RISCOSS builds upon a highly configurable data model that allows customization to several types of scopes. It implements two different working modes: exploration, where the impact of decisions may be assessed before making them; and continuous assessment, where risk variables (and their possible consequences on business goals) are continuously monitored and reported to decision-makers. The blackboard-oriented architecture of the platform defines several interfaces for the identified techniques, allowing new techniques to be plugged in.

Keywords: Open source projects · Open source software · OSS · Open source adoption · Risk management · Software platform

1 Introduction

Risk management is a necessary and challenging task for organisations that adopt open source software (OSS) in their products and in their software development process [1][2]. Risk management in OSS adoption can benefit from the huge amounts of data that are publicly available about the adopted OSS components, as well as data that describes the behavior of OSS communities. The complexity and heterogeneity of the involved data sources, the need to integrate this data with contextual information related to the organisation and its ecosystem, and the convenience of combining different expertise involved in the assessment, call for adequate tools in support of all the phases of risk assessment, from data gathering, to data statistical analysis, to the correlation of these data to the organisational strategic and business risks and assets.

In this paper, we present RISCOSS (www.riscoss.eu), a platform and related assessment methodology for managing risks in OSS adoption [3]. It defines several interfaces to a portfolio of identified measurement and risk management techniques, allowing new techniques to be plugged in if they implement these interfaces and follow well-documented protocols. RISCOSS builds upon a highly configurable data model that allows customization to several types of scopes to support different risk assessment perspectives. It implements two working modes: exploration, where the impact of decisions may be assessed in advance; and continuous assessment, where risk variables (and their consequences on business goals) are monitored, analysed and reported. This allows RISCOSS to support a holistic decision making process inside the adopter organisation.

2 Related Work

Several long-term projects, corporate programs and research initiatives have similar aims than the approach supported by RISCOSS. In Table 1 we summarize the characteristics of the most related European projects that propose methodological approaches and tool support for measuring several aspects of OSS projects, mainly to evaluate the quality of the OSS components and the communities behind them. In particular we refer to the projects with objectives:

- *FLOSSMetrics* (www.flossmetrics.org): constructing, publishing and analysing a large scale OSS database with metrics on OSS development;
- *QualiPSO* (qualipso.icmc.usp.br/OMM/): improving the quality & maturity of OSS projects;
- *QualOSS* (www.libresoft.es/research/projects/qualoss): defining a method to assess the robustness and evolvability of OSS projects;
- *ALERT* (www.alert-project.eu): improving the quality of the software acting on the overall bug resolution process in OSS collaborative environments;
- *OSSMETER* (www.ossmeter.com): supporting the process of discovering, comparing, assessing and monitoring the health, quality, and activity of OSS;
- *MARKOS (*www.markosproject.eu*)*: provides an integrated view on OSS projects, focusing on software functional, structural and license aspects.
- *SQO-OSS* (www.sqo-oss.org): proposing methods and a supporting platform for OSS code quality and community measurement and analysis.
- *U-QASAR* (www.uqasar.eu): providing a quality assurance methodology to assess the quality of software development projects for Internet applications.

All these approaches and platforms focus on the gathering and analysis of OSS data but they are not specifically oriented to inform about the risks derived from these data nor to suggest possible mitigation strategies at the technical and business level.

Table 1. European projects focusing on OSS data analysis

Project	Techniques	Knowledge Models	Tool support
FLOSS Metrics	Databases and analysis techniques to produce OSS project reports	Model of data to describe the characteristics of the different OSS projects	Tools to retrieve data from OSS repositories and produce statistics
QualiPSO	Integration of data from OSS repositories and statistical analysis	A software maturity model with three levels for projects categorization	A platform integrating tools to analyse the source code and the bug tracking systems
QualOSS	Checklist for data retrieving and statistical analysis	A quality model including characteristics, metrics and indicators	Tools to store checklist data and perform analysis of data
ALERT	Integration of data from OSS repositories; statistical analysis and recommendation techniques	Ontologies to support extraction and integration of different data sources	Components able to gather data from OSS sources and services for report visualization and recommendation
OSSMETER	Integration of data from OSS repositories and statistical analysis	Model for OSS forge description; models for the description of OSS quality attributes	Execution of project metrics; storage and analysis of the data and metrics
MARKOS	Integration of data from OSS repositories; license analysis	Ontologies to support the representation of concepts related to code and licenses	Tools to perform code analysis and license conflict analysis
SQO-OSS	Integration of data from OSS repositories and data analysis	Model for OSS quality based on source code and OSS community characteristics	Integration of metrics; analysis of the data through an array of algorithms
U-QASAR	Data gathering on the progress and quality of software development	Models describing software quality and contexts	Platform to obtain an objective value of the software development process quality

Companies and other organizations may also implement similar programs in their development cycles. For instance, we can refer to Codeface, an extensible platform developed by Siemens (http://siemens.github.io/codeface) that aims at gathering data from different OSS sources, to analyse them and to present them to the analyst in a configurable dashboard to support decision-making. The Black Duck suite (https://www.blackducksoftware.com) provides a set of tools for the automated governance and compliance of OSS across the application development lifecycle. From the quality assessment point of view, we can mention the Software Quality Assurance and Trustworthiness (SQuAT) programme at the OW2 consortium aimed at enhancing the perceived reliability of near 50 mature projects in the OW2 code base (http://www.ow2.org/view/About/SQuAT). Some approaches target specific aspects as license assessment, e.g. Palamida (http://www.palamida.com), which provides tools to verify if there is any intellectual property violation in a project adopting OSS; White Source (http://www.whitesourcesoftware.com) provides a solution for companies that need to manage their open source assets to ensure license compliance and reduce risk.

Several recent works face with mining and analysis of OSS projects mainly to assess or predict their quality. For example, in [4] the GHTorent project is presented that had the purpose of collecting data related to different aspects of the quality of the source code for all public projects available on Github. In the area of defect prediction for quality improvement, Peters et al. [5] introduce guidelines to be used in the building of software quality predictors in case of scarcity of data while D'Ambros et al. present a comparison between the different prediction approaches [6]. Zhang et al.

present in [7] a study for the specification of a universal defect predictor. Gamalielsson et al. [8] define the health of an open source ecosystem as an important decision factor when considering the adoption of an OSS component. In [9] the trustworthiness of OSS projects are predicted through the study of the Elementary Code Assessment. RISCOSS can benefit from these studies since it can integrate such quality models and techniques in the risk analysis platform. Adhering to the position defended by Noll et al. [10], RISCOSS calls for human-based qualitative analysis as a necessary component. Some authors define several risks that are associated with adopting an OSS component: project health [11], economic loss and adverse effects on the business processes of the organizations [12], the lack of effective and timely OSS community support for dealing with possible integration problems [13]. Hauge et al. [14] discuss several risks related to OSS adoption and identifies steps for reducing several of these risks. RISCOSS has a holistic perspective that integrates all of these elements in a platform to managing risks related to OSS adoption.

3 RISCOSS Main Functionalities

The main objective of the RISCOSS platform is that of facing the problem of managing risks in a holistic way, supporting the data gathering from the environment of the adopting organisation and from the organisation itself, the analysis of this data for the purpose of OSS risks identification and impact analysis, and the presentation of the data for decision making [3]. Moreover, the RISCOSS platform is designed to support two main operative modes. On the one side, it supports the analyst in performing an explorative analysis and assessment of the OSS ecosystem (in terms of communities and components), for example assessing the convenience of integrating an OSS component in the solution. On the other side, the platform implements a continuous assessment cycle that allows detecting the emerging risks related to OSS choices once, for example, an OSS component has been integrated inside a project.

Based on these basic requirements, some functionalities are proposed which are linked according to the workflow depicted in Fig. 1:

- *Set up of the risk management platform.* When an organization decides to adopt RISCOSS, the platform gathers the needed information to set up a risk management plan (in particular, to determine the key risk indicators), in order to configure the platform infrastructure to the organizational needs. This functionality allows initialising all the knowledge base of RISCOSS having it tailored for the particular organisational environment, including the representation of the business ecosystem where the organisation lives.
- *Identification of the risk assessment level and perspective* (*Elicit scope*). This use case offers the possibility to define a new scope of risk management (see Section 4). Every time one such scope is modified (remarkably, when it is created, e.g. a new project starts, a new OSS component is adopted), it is necessary to set up an organisational view and risk management resources and functionalities for it.
- *Risk assessment.* At any moment, the user may require explicitly risk assessment via situation inspection, what-if analysis by means of e.g. exploration of alternatives, deeper analysis of risk indicators [15], etc. Risk assessment may eventually end up with a change of scope in order to support a holistic risk detection.

- *Reaction to some key risk indicator violation*. As projects evolve and events occur, key risk indicators may be violated. RISCOSS monitors these violations and alerts are triggered when this happens. Then, risk analysis is performed to analyse and eventually solve these situations. Some of the events will be captured by the platform itself by measuring key risk indicators (e.g., a community may be detected to be inactive), some others must be communicated explicitly by decision makers, experts or analysts (e.g., some developer gets a relevant certification or training). Anyhow, this functionality allows for the evolution of the knowledge of the platform following the changes in the organisation and the related ecosystem.

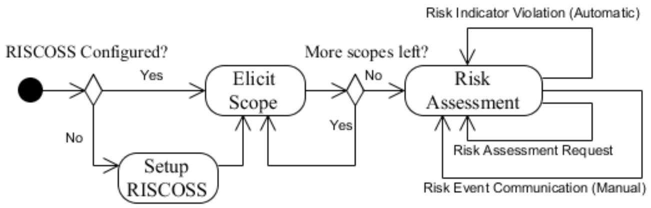

Fig. 1. Workflow for RISCOSS use cases

4 RISCOSS Scopes

We define *scope* as any unit of analysis that can be put under RISCOSS' control. Some organizations may want to monitor the full business; some others may just want to assess the risks related to the adoption of a particular OSS component. In the long term, our platform should be able to cover this entire spectrum. The concept of scope is important to structure the knowledge (and associated actions) managed in RISCOSS. If we refer to risks, every scope may have its different set of risks, e.g. coming from the adoption of some OSS strategy at several levels/scopes. For example, we can have risks as: losing current market position at the level of the organization; not delivering some release in time at the level of the product; involving more resources than expected at the level of the process; exceeding the allocated budget at the level of the project; incorrect selection at the level of the OSS component.

Fig. 2 shows a general view of the concept of scope, its relationships and its reifications, which are currently five (i.e., we have five types of predefined scopes). *Organizational unit*, that wants to supervise a complete portfolio; in a typical organisation, an organisational unit can be seen as a department. *Product*, a commercial good commercialized by the company; it does not necessarily have to be a software product, but of course needs to have some software part that is partially or totally open source. *Process (service)* such as, product manufacturing or product delivering. *Project*, for example, adding a new feature to the current release of a component, or making the necessary steps to deploy a bespoke component in an OSS community. *An OSS component* that is the finest-grained case, and an organization may be interested just in monitoring some adopted OSS component, belonging to a community.

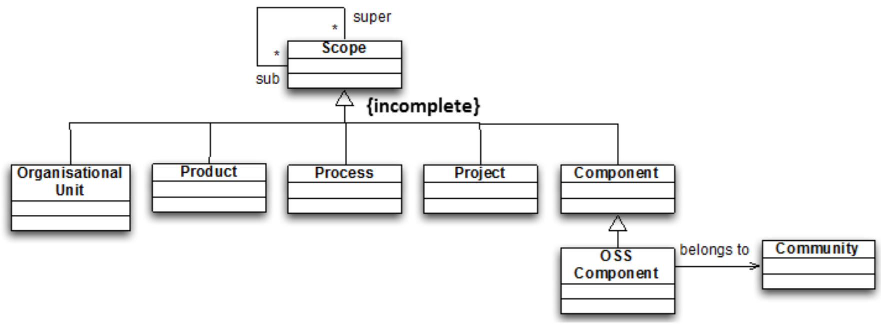

Fig. 2. General RISCOSS scope model

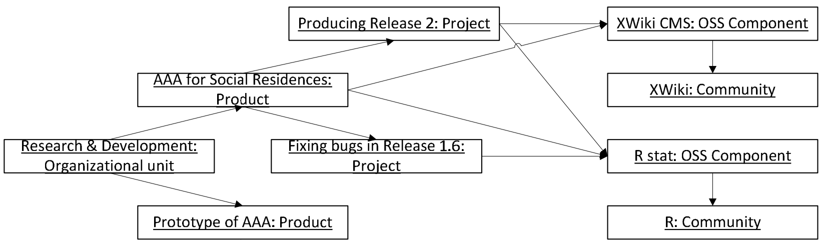
Producing Release 2: Project
XWiki CMS: OSS Component
AAA for Social Residences: Product
XWiki: Community
Research & Development: Organizational unit
Fixing bugs in Release 1.6: Project
R stat: OSS Component
Prototype of AAA: Product
R: Community

Fig. 3. An instantiation of the RISCOSS scope model

The class diagram shows how scopes are highly configurable, because: 1) the specialization is "incomplete", so that new types of scope may be added; 2) the reflexive association allows to build arbitrary hierarchies of scopes, so that in one organization, a project may include products, whilst in other a product may include projects.

In Fig. 3, we show an instantiation of the scope structure in which the Research & Development Unit of a Research institution has an OSS business strategy in place for the production of research prototypes and pre-competitive products to be tested on the field. The organization is focusing on two products: "Ambient Aware Assistance" (AAA), an environmental monitoring platform, and "AAA for Social Residences" (AAASR) that is a system for monitoring health conditions of patients living in social residences. The organization is currently running two projects in parallel for AAASR, "Producing release 2" and "Fixing bugs in release 1.6". The products are mixing different types of components, and for OSS components it adopts "XWiki CMS", an OSS web content management system developed by the "XWiki community", and "R stat", a complex statistical package, deployed by "R community". This second component is used only in the release 2. The Research institution would also like to affect "R community" releasing new statistical procedures to the community. Moreover, the organisation wants to restrict the scope of analysis of projects' components to those that are part of some commercialized product, i.e. if an OSS component is just used as part of the project management (e.g., a version control management tool) the organization is not interested in it.

5 Tool Architecture

The section introduces the basic RISCOSS platform architecture (see Fig. 4). The central element is a content management tool, XWiki, which is the unifying element of the platform and covers three basic responsibilities: (i) Offering an interface to the user for dialoguing with the RISCOSS platform: as such, XWiki offers and organizes the required forms and dashboards, maintains documents, and supports, as a wiki tool, collaborative work as needed. (ii) Integrating other tools that perform functionalities required by the platform: XWiki is the umbrella that coordinates these tools, gathering the results of their computation to feed internal data structures, accessible from the tools in a blackboard architecture fashion, and allowing other tools to initialise their calculations. (iii) Accessing the reusable knowledge of the platform, namely the model patterns, data, form templates created by experts. The platform defines families of tools that are integrated into XWiki by means of well-established interfaces:

- *Questionnaire tool.* The tool gathers from decision-maker and experts of a given organisation the information related to the characteristics of the organisation and the initial scopes.
- *Goal and risk modelling tools.* The questionnaires are used, among other things, to create models that represent the ecosystem of the organization, with organizational goals stated using *i** [16], and risks models bound to them considering risks that are related to software quality, community behaviour or OSS licenses. This allows making explicit the consequences of risks in the OSS ecosystem.
- *Risk reasoning engines.* In particular:
  - *Logical reasoning tools.* These tools perform model analysis and reasoning in order to allow for risk identification and management. The reasoning tool currently implements model label propagation [17] techniques and disjunctive logic algorithms [18] in order to identify and mitigate the risks that can occur given specified environmental situations
  - *Statistical reasoning tools.* RISCOSS relies, among others, on the implementation of Bayesian Network based components that are used to reason about the correlation between the measures obtained by the analysis of the OSS communities and the strategic and business risks of the OSS adopters. These tools rely on a strong interaction with experts and analysts to allow for an assessment and revision of the correlations between the identified measures and the strategic and business risks.
- *Measuring and monitoring framework* composed of several Risk Data Collectors. These tools measure risk indicators and feed XWiki with the obtained values, and also monitor, detect and communicate anomalous situations from the risk management point of view. To do so, this framework needs to: 1) obtain data from the selected data sources (such as license or code quality analysis tools or blogs, forums and mailing lists of the communities); 2) implement basic statistical analysis, for example to compute statistical distributions of data [19], analysis of OSS communities structure and behaviour [20],
- *OSS Risk Dashboard.* This element collects the output of the risk assessment process to visualise it in a single view summarising the main aspects and giving the possibility to enter in the details of the single information. Here we envisage means to show the risk exposure of the organisation with respect to, for example, the adoption of a particular OSS component, or the result of a *what-if* analysis process.

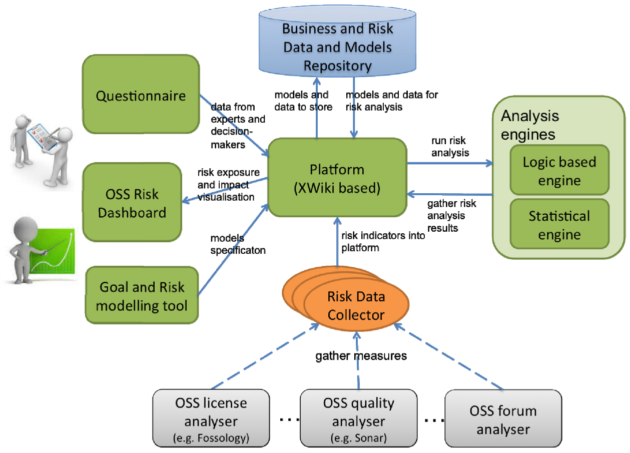
Business and Risk Data and Models Repository
Questionnaire
data from experts and decision-makers
models and data to store
models and data for risk analysis
Analysis engines
Logic based engine
Statistical engine
run risk analysis
gather risk analysis results
Platform (XWiki based)
OSS Risk Dashboard
risk exposure and impact visualisation
models specificaton
Goal and Risk modelling tool
risk indicators into platform
Risk Data Collector
gather measures
OSS license analyser (e.g. Fossology)
OSS quality analyser (e.g. Sonar)
OSS forum analyser

Fig. 4. RISCOSS platform logical view

In addition to these tools we can envision some other types to be added later in the platform, for instance, estimation cost tools to bind risks to project cost estimation.

It is worth to mention that XWiki manages a repository of reusable knowledge (the platform knowledge base). In this repository, we store the artifacts that can be reused for future and continuous assessment, such as patterns for goal and risk models.

Fig. 5 shows a component-oriented view that highlights the main type of components present in the RISCOSS platform and how they are interconnected.

The RISCOSSPlatform-XWiki component implements the RISCOSSPlatform API by using the APIs provided by XWiki. This component provides all the business logic for operating the RISCOSS Platform on top of the XWiki Platform. It requires some tools implementing the Tool API in order to perform the actual analysis. In particular these tools are the way for extending the RISCOSSPlatform with new capabilities, e.g. new risk analysis tools. A Tool component, on the other hand, provides the Tool API interface and requires the ToolConfigurationProvider API interface. This allows the tool to ask, in a standard way, configuration parameters that might be needed for its initialization. The RISCOSSPlatform-XWiki component is responsible to manage the persistence and retrieval of this information. This will be managed using the XWiki Platform that will store this information and will also provide a dedicated UI for manipulating it. The RISCOSSPlatform implementation is made available to the XWiki Platform via a component implementing the ScriptService API. This component will expose the relevant methods of the RISCOSSPlatform API to the UI component of the XWiki Platform that allows invoking the functionalities of the RISCOSS Platform in order to provide to the end user a user interface through which interact with it. The user interface will be also used to show the status and the results of the analysis performed, and the content of the collected information (i.e., the RISCOSS Knowledge base). The Storage component will provide the means for storing data persistently in a DBMS that supports the JDBC API.

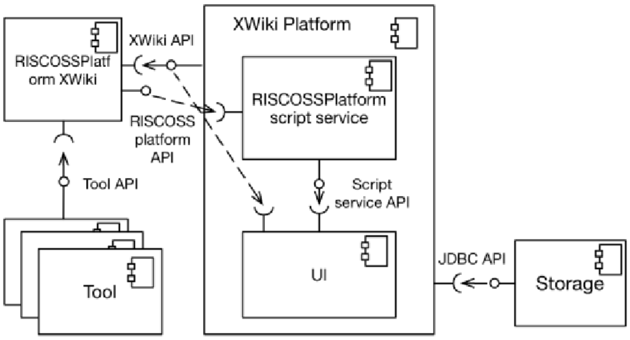

Fig. 5. RISCOSS platform deployment view

6 Discussion

The platform is currently in active evolution. The part already implemented consists of the set of main components for the different tool orchestration and a set of user interaction and risk and business analysis tools. The risks currently addressed in the platform are those related to OSS licenses violatoin and to software quality. The architecture is highly modular and allows for the definition of several possible tools that with the help of the coordination of the XWiki CMS platform can easily implement other services for the analyst. In particular, a possible direction of evolution is that of increasing the possibility of the decision-maker to interact with the reasoning engines via questionnaires and the implementation of new reasoning engines able to exploit this user knowledge in a more interactive way.

In the line of facilitating the use of the platform in different organizational settings from OSS communities to small and large companies adopting OSS, several deployment schemas have been considered, from a web-based installation where the different users that can create their own workspace and, at the same time use the common knowledge collected by the platform from the different users, to an organization specific installation, that can be also web-based, but that is confined in the premises of the organization. The first solution implicitly allows for another point of extension that is the creation of a large and evolving Business and Risk models knowledge base that can be reused and extended by other analysts, so promoting the creation of a community around the building of new knowledge about risks.

7 Conclusions

In the paper we presented the RISCOSS platform supporting risk assessment activities in OSS adoption. The peculiar aspect of the platform is that it aims at supporting the whole decision making process: from the gathering of the data from heterogeneous sources, to the risk indicator identification, to the modelling and assessment of the impact of the risks to the strategic and business goals of the organisation. We are currently deploying the platform in several contexts (large company with complex organizational structure, public administration, SME IT provider, and a community hosting ca. one hundred OSS projects) to improve its functionalities and knowledge.

Acknowledgement. This work is a result of the RISCOSS project, funded by the EC 7th Framework Programme FP7/2007-2013, agreement number 318249.

References

1. Li, J., et al.: A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf Software Components. IEEE TSE **34**(2) (2008)
2. Hauge, Ø., Cruzes, D.S., Conradi, R., Velle, K.S., Skarpenes, T.A.: Risks and risk mitigation in open source software adoption: bridging the gap between literature and practice. In: Ågerfalk, P., Boldyreff, C., González-Barahona, J.M., Madey, G.R., Noll, J. (eds.) OSS 2010. IFIP AICT, vol. 319, pp. 105–118. Springer, Heidelberg (2010)
3. Franch, X., et al.: Managing risk in open source software adoption. In: ICSOFT 2013 (2013)
4. Gousios, G.: The GHTorent dataset and tool suite. In: MSR 2013 (2013)
5. Peters, F., Menzies, T., Marcus, A.: Better cross company defect prediction. In: MSR 2013 (2013)
6. D'Ambros, M., Lanza, M., Robbes, R.: An extensive comparison of bug prediction approaches. In: MSR 2010 (2010)
7. Zhang, F., Mockus, A., Keivanloo, I., Zou, Y.: Towards building a universal defect prediction model. In: MSR 2014 (2014)
8. Gamalielsson, J., Lundell, B., Lings, B.: The nagios community: an extended quantitative analysis. In: Ågerfalk, P., Boldyreff, C., González-Barahona, J.M., Madey, G.R., Noll, J. (eds.) OSS 2010. IFIP AICT, vol. 319, pp. 85–96. Springer, Heidelberg (2010)
9. Lavazza, L., Morasca, S., Taibi, D., Tosi, D.: Predicting OSS trustworthiness on the basis of elementary code assessment. In: ESEM 2010 (2010)
10. Noll, J., Seichter, D., Beecham, S.: A qualitative method for mining open source software repositories. In: Hammouda, I., Lundell, B., Mikkonen, T., Scacchi, W. (eds.) OSS 2012. IFIP AICT, vol. 378, pp. 256–261. Springer, Heidelberg (2012)
11. Piggot, J., Amrit, C.: How healthy is my project? open source project attributes as indicators of success. In: Petrinja, E., Succi, G., El Ioini, N., Sillitti, A. (eds.) OSS 2013. IFIP AICT, vol. 404, pp. 30–44. Springer, Heidelberg (2013)
12. Petrinja, E., Sillitti, A., Succi, G.: Comparing OpenBRR, QSOS, and OMM assessment models. In: Ågerfalk, P., Boldyreff, C., González-Barahona, J.M., Madey, G.R., Noll, J. (eds.) OSS 2010. IFIP AICT, vol. 319, pp. 224–238. Springer, Heidelberg (2010)
13. Ayala, C., Cruzes, D.S., Nguyen, A.D., Conradi, R., Franch, X., Höst, M., Babar, M.A.: OSS integration issues and community support: an integrator perspective. In: Hammouda, I., Lundell, B., Mikkonen, T., Scacchi, W. (eds.) OSS 2012. IFIP AICT, vol. 378, pp. 129–143. Springer, Heidelberg (2012)
14. Ligaarden, O.S., Refsdal, A., Stolen, K.: ValidKI: a method for designing key indicators to monitor the fulfillment of business objectives. In: BUSTECH 2011 (2011)
15. Yu, E.S.K.: Modelling Strategic Relationships for Process Reengineering. PhD thesis, University of Toronto, Canada (1995)
16. Nilsson, N.J.: Problem-solving Methods in Artificial Intelligence. McGraw-Hill (1971)
17. Leone, N., et al.: The DLV System for Knowledge Representation and Reasoning. ACM Transactions on Computer Logic **7**(3) (2006)
18. Kenett, R.S., Zacks, S.: Modern Industrial Statistics: with applications in R, MINITAB and JMP, 2nd (edn.). John Wiley and Sons (2014). With contributions by D. Amberti
19. Salter-Townshend, M., White, A., Gollini, I., Murphy, T.B.: Review of Statistical Network Analysis: Models, Algorithms, and Software. Stat. Analysis and Data Mining **5**(4) (2012)