Mora, M., Marx Gómez, J., O'Connor, R.V. and Gelman, O. (2016) 'An MADM risk-based evaluation-selection model of free-libre open source software tools', Int. J. Technology, Policy and Management, Vol. 16, No. 4,

A MADM Risk-based Evaluation-Selection Model of Free-Libre Open Source Software Tools

Manuel Mora
Information Systems Department, Autonomous University of Aguascalientes, Mexico
Email: mmmora@securenym.net

Jorge Marx Gómez
Department of Informatics, University of Oldenburg, Oldenburg, Germany,
Email: jorge.marx.gomez@uni-oldenburg.de

Rory V. O'Connor
School of Computing, Dublin City University, Ireland,
Email: rory.oconnor@dcu.ie

Ovsei Gelman
CCADET, Universidad Nacional Autónoma de México, Mexico,
Email: ogelman@unam.mx

Abstract: Free-Libre Open Source Software (FLOSS) tools are free-cost license highly attractive to be implemented by organizations. However, not of all the FLOSS tools are mature, and failed implementations can occur. Thus, FLOSS evaluation-selection frameworks and FLOSS success-failure implementation factors studies have been conducted. In this research, we advance on such studies through an integrated FLOSS evaluation-selection model with a risk-based decision making approach. Our model was built upon the other two literatures, and it was structured as a Multi-Attribute Decision Making (MADM) model which contains 12 variables grouped in four risk categories: financial, organizational, end-user and technical ones. We illustrated its utilization in the domain of Information Technology Service Management (ITSM) FLOSS tools. Hence, our model contributes to the FLOSS literature with the inclusion of the risk management approach and to the FLOSS evaluation-selection praxis with the provision of an innovative and essential risk-based model.

Keywords: FLOSS evaluation; FLOSS implementation; risk management; MADM; value tree; IT service management.

Reference to this paper should be made as follows: Mora, M., Marx Gómez, J., O'Connor, R.V. and Gelman, O. (2016) 'An MADM risk-based evaluation- selection model of free-libre open source software tools', Int. J. Technology, Policy and Management, Vol. 16, No. 4, pp.326–354.

Biographical notes: Manuel Mora is a Professor in the IS Department at the Autonomous University of Aguascalientes (UAA), Mexico. Dr. Mora holds a B.S. in Computer Systems Engineering (1984) and M.Sc. in Computer Sciences (1989) from Monterrey Tech (ITESM), and an Eng.D. in Engineering (2003) from the National Autonomous University of Mexico (UNAM). He has published over 80 research papers in international top conferences, research books, and research journals. Dr. Mora is ACM senior member, SNI Level I, and EiC of IJITSA journal, and serves also in the ERBs of several international journals focused on DMSS and IT services. His current research streams are: DMSS, ITSM and Complex Systems.

Jorge Marx Gómez is full-time Professor and Chair of Business Information Systems at the Carl von Ossietzky University of Oldenburg, Germany. He holds a Bachelor in Computer Engineering and Industrial Engineering at the University of Applied Science of Berlin (Technische Fachhochschule), and PhD in Business Information from Otto-von-Guericke-Universität Magdeburg. His main personal research interests are: environmental management information systems, business intelligence, enterprise systems engineering, and business information systems in higher education. He is a member of: German Association of Computer Science (Gesellschaft für Informatik e.V., in short GI), OFFIS e.V. (Oldenburg Institute of Informatics), SAP Roundtable for Business Intelligence, German Association of University Professors and Lecturers (Deutscher Hochschulverband, in short DHV), German Forum of Interoperability (in short DFI), German Oracle Users Group e.V. (in short DOAG) and reviewer and expert in DAAD selection committee for Latin-American research proposals.

Rory O'Connor is a Professor of Computing at Dublin City University, where he is currently serving as the Head of School of Computing. He received a PhD. in Computer Science from City University, London (2000), a M.Sc. in Computer Applications from Dublin City University (1995), a B.Sc. in Computer Applications from Dublin City University (1993) and a National Certificate in Computing, Waterford Regional Technical College (1990). He is also a member of both the Irish Computer Society and British Computer Society. He has previously held research positions at both the National Centre for Software Engineering and the Centre for Teaching Computing. In addition, he worked with a variety of organization's, undertaken software development projects, conducting software quality reviews and inspections. He was a visiting lecturer to the University of Iceland in the 2007/8 academic year. His research interests are centered on the processes whereby software intensive systems are designed, implemented and managed. In particular, researching methods, techniques and tools for supporting the work of software project managers and software developers in relation to software process improvement, software project planning and management of software development projects.

Ovsei Gelman-Muravchik is a Senior Researcher at the Center of Applied Sciences and Technology Development (CCADET) of the National Autonomous University of Mexico (UNAM). He holds a B.S., a M.Sc. and a Ph.D. in Physics and Mathematics from the University of Tbilisi, Georgia. In the last 40 years. Dr. Gelman-Muravchik has contributed to the advance of the Systems Science discipline, through the publication of approximately 250 research papers in books, national and international journals and conference proceedings, the conduction of consulting activities for governmental and private organizations and the participation as advisor in the Graduate University Program of Engineering and Business Administration at UNAM. Professor Gelman-Muravchik was a founding member of the Editorial Council of the Yearbook "Systems Research" from 1969 till 1972, published by the

Public Council established in 1968 for promotion and development of Systems Research in Soviet Union.. As well, from 1964 till 1972, he was the Scientific Editor of the Cybernetics Institute publications of the Georgian Academy of Sciences; from 1996, during several years, a member of the Honorary Editorial Advisory Board of the Encyclopedia of Life Support Systems (EOLSS) and from 2002 till 2006, he was the Editor-in-Chief of the "Journal of Applied Research and Technology&", published by CCADET-UNAM.

1 Introduction

The free-libre open source software (FLOSS) is defined essentially as a software product that has a free-cost license of use, whose source code is available and it can be modified, and should be not linked a specific IT technology (Coar, 2005). According to Androutsellis-Theotokis et al. (2011), the origins of FLOSS can be traced to the SHARE, Unix and GNU projects. FLOSS has gained organizational acceptance after the successful utilizations of high quality FLOSS star products like Linux operating system, Apache web server, MySQL data base management system, OpenOffice suite, PHP language, Mozilla Firefox browser, Sendemail mailer tool, and R tool for statistics, among others (Nagy et al., 2010; Androutsellis et al., 2011). Other acceptance reinforcement has been provided by large and well-recognized IT firms like IBM, Sun, and Google that have promoted the creation of FLOSS communities or have promoted the utilization of some FLOSS products (Nagy et al., 2010; Spinellis and Giannikasa, 2012). For instance, IBM supports the Java Eclipse platform, Sun the Netbeans platform, and Google the massive utilization of Linux Operating Systems (Spinellis and Giannikasa, 2012). In particular, the organization SourceForge.net keeps a database over 130,000 FLOSS projects.

According to Watson et al. (2008), the FLOSS phenomenon breaks typical barriers on acquisition and distribution costs, as well as physical and legal frontiers through its simple access via Internet. Thus, their high availability has stimulated their organizational implementation in several developed countries (David et al., 2003). However, FLOSS products cannot be considered totally free-cost implementations. Nagy et al. (2010) identified hidden costs implied with the utilization of FLOSS products –despite they are per se free-cost- such as: user training costs as any new software, technical implementation and operating support costs; and integration costs with legacy systems. Furthermore, it has been also reported that FLOSS products are less user-oriented polished products than proprietary alternatives, and the availability of unsupported multi-versions (i.e. the forking problem) can cause FLOSS implementation failures in organizations. Thus, organizations interested in the utilization of FLOSS products must conduct a careful evaluation-selection process (idem, 2010) and this represents a problem to Information Technology (IT) managers. A wrong selection of a FLOSS tool, from the usual extensive variety of them that is available- will produce negative effects as any failed IT implementation (Ven et al., 2008).

For this aim, several FLOSS evaluation-selection frameworks have been reported in the literature (Nagy et al., 2010; Aversano and Tortorella, 2013). Some of them (for instance Navica (Golden, 2005) and QMOSS (Sung et al., 2007)) are simple models composed for 5-7 single factors and 3-5 steps, and others are complex ones with over 10-15 steps and 60 evaluation items (QualiPSo (del Bianco et al., 2009) and QSOSv2 (QSOS.org, 2013)). Additionally, other studies have identified a set of organizational factors associated to successful and failed utilizations of FLOSS tools in organizations (Dedrick and West, 2003; Rossi et al., 2012; Li et al., 2013), which provide also useful information to avoid failed and wrongly FLOSS evaluations. Hence, while FLOSS has been used for large companies, we consider that FLOSS is a natural logical resource to be used for small and medium-sized companies, which have more budgetary and human resource limitations for using commercial software (Tribunella and Baroody, 2007). However, despite of the availability of FLOSS evaluation frameworks and FLOSS implementation studies, we identify that this cumulated knowledge is disperse, fragmented and complex to be applied by small organizations.

Thus, in this research we pursue the goal to design a simple and theoretically supported FLOSS evaluation-selection model suitable for small organizations. For this aim we review, at first, two core literatures: on FLOSS evaluation-selection models and on FLOSS implementation models. We enhance this design through an innovative risk-based approach (Stoneburner et al., 2002). The FLOSS evaluation-selection model is implemented with a multi-attribute decision making mechanism (MADM) (Yoon and Hwang, 1995; Roy, 2005). We illustrate this model with the evaluation-selection of a FLOSS tool in the domain of ITSM (Gallup et al., 2009; Brenner, 2006). This paper is an extended and enhanced version of an initial research already reported by authors (Mora et al., 2015).

The remainder of this paper continues as follows: in section 2, we review the both core FLOSS literature on evaluation frameworks and implementation factors; in section 3, we report the design of the MADM FLOSS evaluation-selection model; in section 4, we illustrate its utilization with a demonstrative realistic case of evaluation-selection of a FLOSS ITSM tool; finally, in section 5, we report conclusions, limitations and recommendations for further research.

2 Theoretical Background

2.1 Review of Literature on FLOSS Frameworks

Several frameworks have been reported (Nagy et al., 2010; Stol and Babar, 2010; Aversano and Tortorella, 2013) for evaluating and selecting FLOSS products. These models consider factors such as (Nagy et al., 2010) the availability of training, documentation, third party support, integrated software and other professional services, community size, community age, and lines of source code. These factors are used in simple decision-making models with different weights for each factor, to estimate the maturity of open source software. These frameworks, thus, consider not only the software per se but additional factors (developer community, general user community, organizational attributes), but their decision-making mechanism used can be considered simple (Yoon and Hwang, 1995; Roy, 2005).

Quality issues directly related with the software product have been based mainly in the ISO/IEC 9126 standard (ISO, 1991). In general, quality is defined as the "*degree to which a set of inherent characteristics fulfils requirements*" (ISO, 2005; p.7). The ISO/IEC 9126 defines six measurable attributes for assessing the overall quality of software: functionality, reliability, usability, efficiency, maintainability and portability. Functionality refers to extent of the software provides the required functions for the intended user. Reliability refers to the extent of the software is used without failures. Usability refers to the extent of the software is ease to use. Efficiency refers to the extent of the software has a congruent performance behavior regarding its used resources. Maintainability refers to the extent of the software is ease of modification and upgrade. Finally, portability refers to the extent of the software can be transferred to other platform. This ISO/IEC 9126 has been updated to a new ISO/IEC 25010:2011 standard (SQuaRE) (ISO, 2011). Both ISO/IEC frameworks have been posed for software products in general. However, according to Aversano and Tortorella (2013), the

FLOSS products contain additional characteristics –due to its different community development process- and thus adaptations of these quality software evaluation models must be applied. Specific FLOSS evaluation frameworks, thus, have been proposed focused on the product, the development process or both elements.

We found 12 FLOSS evaluation frameworks in the literature: Capgemini Open Source Maturity Model (Duijnhouwer and Widdow, 2003), Navica Open Source Maturity Model (OSMM) (Golden, 2005), Open Business Readiness Rating (OpenBRR) (OpenBRR.org, 2005), Open Business Quality Rating (OpenBQR) (Taibi et al., 2007), Quality Model for Open Source Selection (QMOSS) (Sung et al., 2007), QualOSS (Deprez, 2008), Software Quality Observatory for Open Source Software model (SQO-OSS) (Samoladas et al., 2008), OpenSource Maturity Model (OMM) (Petrinja et al., 2009), QualiPSo—Quality Platform for Open Source Software (del Bianco et al., 2009), IRCA Model (Wheeler, 2011), Method for Qualification and Selection of Open Source Software (QSOSv2) (QSOS.org, 2013), and the Evaluation Framework for Free/Open Source Projects (EFFORT) (Aversano and Tortorella, 2013). We analyzed carefully these 12 FLOSS evaluation frameworks for: 1) assessing their overall suitability for being applied in IT areas of small business, and 2) identifying the shared criteria structure. The Table 1 and 2 report respectively the results for the analyses 1) and 2).

In analysis 1), the following elements were considered: framework, year of publication, focus and scope, structural complexity, functional complexity, availability of public documentation, tool support, risk management inclusion, and suitability for small business. Focus and scope refers to the entities (product, organization) included in the evaluation and the type of FLOSS (general or particular). Structural complexity is defined in this research as the extent of the model presents a low, moderate or high conceptual density structure of criteria. Functional complexity is defined in this research as the extent of the model presents a low, moderate or high procedural difficulty of applying the steps proposed in the model. Lightweight process refers to the extent of agility to apply the evaluations steps. They might ease of applying but being to numerous becoming the process in a heavy one. Availability of public documentation refers to the free-cost documentation on the utilization of the model. Tool support refers to the availability of a FLOSS software for applying the model. Risk management inclusion refers to the explicit consideration and inclusion of risk management practices into the model. Finally, suitability for small business refers to an overall recommendation on the economic and organizational feasibility of being used in small business based on the previous criteria.

Our analysis reported in the Table 1 reveals the following insights:

1. all of the FLOSS evaluation frameworks consider both the software product and the development organization;
2. almost FLOSS evaluation frameworks (11 of the 12) addresses generic FLOSS;
3. while there are seven FLOSS evaluation frameworks assessed with low or moderated structural and functional complexity, only three of them presented a lightweight process suitable for small organizations (Navica Open Source Maturity Model (OSMM), Quality Model for Open Source Selection (QMOSS), and OpenSource Maturity Model (OMM);
4. the three FLOSS evaluation frameworks suitable for small organizations provide public information but it is minimal;

5. just one (Navica Open Source Maturity Model (OSMM)) of the three FLOSS evaluation frameworks provides a tool support via templates;
6. five of twelve FLOSS evaluation frameworks provide a web-based tool but it is for private use (Open Business Quality Rating (OpenBQR), QualOSS, Software Quality Observatory for Open Source Software model (SQO-OSS), QualiPSo—Quality Platform for Open Source Software, and Method for Qualification and Selection of Open Source Software (QSOSv2); and
7. only one of the 12 FLOSS evaluation frameworks uses explicitly a risk management approach (QualOSS) but it is not suitable for small and medium-sized business.

Table 1. Analysis of FLOSS frameworks

FLOSS Framework	Year	Focus and scope	Structural complexity	Functional complexity	Light-weight process	Public documentation	Use of risk management	Overall suitability for SMBs
CapGemini	2003	Product/Process	Moderate	Moderate	No	No	Implicit	Moderate
Navica	2004	Product/Process	Low	Low	Yes	Minimal	Implicit	High
OpenBRR	2005	Product/Process	Moderate	Moderate	No	Moderate	Implicit	Moderate
OpenBQR	2007	Product/Process	Moderate	Moderate	No	Moderate	No reported	Moderate
QMOSS	2007	Product/Process	Low	Low	Yes	Minimal	No reported	High
QualOSS	2008	Product/Process	High	High	No	Moderate	Explicit	High
SQO-OSS	2008	Product/Process	High	High	No	Minimal	No reported	Low
QualiPSo	2009	Product/Process	High	High	No	Minimal	No reported	Low
OSMM	2009	Product/Process	Low	Low	Yes	Minimal	No reported	High
IRCA	2011	Product/Process	Moderate	High	No	Moderate	Implicit	Moderate
QSOSv2	2013	Product/Process	High	High	No	Moderate	Implicit	Low
EFFORT	2013	Product/Process	High	High	No	Moderate	No reported	Low

Table 2. Summary of FLOSS evaluation shared evaluation attributes

FLOSS Framework	Product attributes																	Organization attributes					
	Correctiveness	Customizability	Documentation	External reviews	Functionability - Quality	Interoperability - portability	Licensing - legal issues	Maintainability	Marketing - popularity	Maturity - longevity	Performance - efficiency	Standard compliance	Security - reliability	Test information	Usability	User community	User satisfaction	Developer community	Developer process	Organizational structure	Support	Technical environment	Training
CapGemini	~	●	~	~	●	●	●	●	●	●	~	●	~	~	~	●	~	●	~	●	●	~	~
Navica	~	~	●	~	●	●	~	~	~	●	~	~	~	~	~	~	~	~	~	~	●	~	●
OpenBRR	~	●	●	●	●	●	●	●	~	●	●	●	●	~	●	●	~	●	~	●	●	~	~
OpenBQR	~	~	●	●	●	~	●	~	~	~	~	●	~	~	●	●	~	●	~	●	~	~	●
QMOSS	~	~	~	~	●	~	~	~	~	~	~	~	●	~	●	~	~	~	~	~	~	~	~
QualOSS	●	~	●	~	~	~	~	●	~	~	~	~	●	●	~	~	~	●	●	●	~	~	~
SQO-OSS	~	~	●	~	●	~	~	●	~	●	~	~	●	●	~	~	~	●	~	~	~	~	~
QualiPSo	~	~	~	~	●	●	~	●	~	~	●	~	●	~	~	~	●	●	~	~	~	~	~
OSMM	~	~	●	●	~	~	●	●	●	●	~	●	~	●	~	●	~	~	●	~	●	●	~
IRCA	~	~	~	~	●	~	~	●	~	●	●	~	●	~	●	~	~	~	~	~	~	~	~
QSOSv2	~	~	●	~	●	~	●	●	●	●	~	~	~	●	~	●	~	●	●	●	●	~	●
EFFORT	●	●	●	~	●	●	●	●	●	●	●	●	~	●	●	●	~	●	~	●	●	~	●
Frequency	**2**	**3**	**8**	**3**	**10**	**5**	**6**	**9**	**4**	**8**	**4**	**5**	**6**	**5**	**4**	**6**	**1**	**8**	**3**	**6**	**6**	**1**	**4**

In analysis 2), we identified 17 attributes for the software product and 6 for the development organization. The Table 2 summarizes these findings. Interesting finding is that none of the total 23 attributes was reported in all of the 12 FLOSS frameworks. The frequency found of attributes was from 1 to 10. For the software product, the most reported attributes (at least in 50% of the FLOSS frameworks) were the following: functionality-quality (10 times), maintainability (9 times), maturity-longevity and documentation (8 times), secure-reliability, user community and licensing-legal issues (6 times). Regarding the less frequent attributes for the software product these were the following: external reviews and maturity (3 times), cost effectiveness (twice), and user satisfaction (once).

For the development organization, the attribute most reported (at least in 50% of the FLOSS frameworks) were the following: developer community (8 times), and licensing-legal issues, support, organizational structure and user community (6 times). The less frequent attribute was: technical environment (1 time). Hence, this literature review on main FLOSS evaluation frameworks has provided useful insights regarding the: suitability for being applied in small organizations by being lightweight processes; the set of most and less reported evaluation attributes in the FLOSS evaluation frameworks; and the finding on the lack of risk management approaches.

2.2 Review of Studies on FLOSS Success-Failure Implementation Factors

In the initial period of 2003-2005 several FLOSS implementation studies have been reported (Dedrick and West, 2003; Fitzgerald and Kenny, 2004; Goode, 2005; Holck et al., 2005; Waring and Maddocks, 2005; Verma et al., 2005). In the first study of Dedrick and West (2003), it was investigated the reasons for implementing FLOSS in ten organizations through a qualitative data-grounded theory-building method. As theoretical lenses for collecting and organizing data, they used a TOE framework (technology, organization, and environment) (Depietro et al., 1990). Factors such as: no licenses costs, new business opportunities, functionality, reliability, ease of use, and compatibility, were found enablers for successful FLOSS implementations. In turn, lack of internal expertise in FLOSS tool, and lack of external support were found as inhibitors for successful FLOSS implementations.

Fitzgerald and Kenny (2004) investigated the transition of a large Irish hospital from proprietary software to FLOSS. This hospital did a change of software platforms (financial system, email system, server application systems, among others) for facing IT budget reductions. Investigators (idem, 2004) found the following enablers existent for this shift: no license costs, avoidance of proprietary lock-in, top management support, and user involvement. Regarding FLOSS implementation inhibitors that were avoided these were the following: IT staff resistance, lack of supplier responsibility. Important economic savings were also reported by investigators. Goode (2005) surveyed 500 Australian top firms for investigation FLOSS implementation inhibitors. This due to the low FLOSS adoption rate. Goode (2005) found the following inhibitors: IT staff resistance, switching costs (and training costs), lack of supplier responsibility, and lack of relevance.

Holck et al. (2005) reviewed literature on FLOSS implementation factors for identifying enablers and inhibitors jointly with conducted case studies in Danish organizations. Investigators found the following enablers: no license fees, functionality-quality, and compatibility. For the case of inhibitors, the investigators found: lack of internal expertise, lack of external support, and lack of supplier responsibility. Waring and Maddocks (2005) investigated 8 case studies in UK public governmental organizations. Authors (idem, 2005) found the following enablers: no license costs, functionality-quality, reliability, and customizability. None inhibitor was reported. Verma et al. (2005) surveyed two Linux user communities at USA and India to identify factors for adopting FLOSS (in this case the Linux operating system). Investigators tested several factors (voluntariness, usefulness, compatibility, image, ease of use, result demonstrability, visibility and trialability) and found surprisingly that only compatibility and ease of use were associated to the adoption of FLOSS. These results can be explained due to the type of community: highly technical specialized one rather end-user community.

Additional studies from 2006-2013 period complement the findings on FLOSS implementation enablers and inhibitors (Ven and Verelst, 2006; Morgan and Finnegan, 2007; Gallego et al., 2008; Sohn and Mok, 2008; Lee et al., 2009; Hauge et al., 2010; Nagy et al., 2010; Rossi et al., 2012; Li et al., 2013). In Ven and Verelst's study (2006), five case studies are reported from Belgian organizations that adopted the utilization of a FLOSS operating system. Two were large (at least 500 employees), one medium-sized (among 100-500 employees) and two small organizations (at most 100 employees). Investigators found the following enablers: no license fees, avoidance of proprietary lock-in, reliability, employee skill and training, trialability, and standard compliance. These authors (idem, 2006) also accounts as inhibitors (which were avoid by these 5 organizations) with: switching costs, lack of internal support, and lack of external support. In next study from Morgan and Finnegan (2007) it was investigated 13 organizations in the software industry located in Europe. In general, the investigators found several technology, organizational, environmental and individual enablers and inhibitors of successful FLOSS implementations. Main enablers identified were: no license costs, avoidance of proprietary lock-in, new business opportunities, functionality-quality, performance-efficiency, reliability-security, compatibility, standard compliance and implementation champion. Regarding the inhibitors they were the following ones: lack of internal expertise, lack of external expertise, hard selection process, and lack of documentation. Gallego et al's study (2008) surveyed European communities of FLOSS Linux users. They found enablers such as: functionality-quality, customizability-portability (flexibility), ease of use, and usefulness-relevance. Their study did not investigate inhibitors. In next study Sohn and Mok (2008) surveyed a population of Korean programmers working at Korean organizations. They found the following enablers: functionality, efficiency, reliability, portability and sharing knowledge between communities.

In next study of Lee et al. (2009) surveyed international Linux communities. They found the following enablers: quality, community support, expert developer community, and usefulness (satisfaction). Inhibitors were not studied. In Hauge et al. (2010), a large Norwegian telecommunication was investigated to identify enablers and inhibitors (considered explicitly as risks). Investigators found the following enablers: no license fees, avoidance of proprietary lock-in, community support, employee skill, top management, and trialability. The identified inhibitors (reported as risks) were: lack of internal support, lack of external support, hard selection process, switching costs (and hidden costs), lack of supplier responsibility, and uncontrolled utilization. Nagy et al.'s study (2010) was focused on FLOSS implementation inhibitors (called barriers). They identified in their conceptual study the following enablers also: no license fees and avoidance of proprietary lock-in. The main found inhibitors were: switching costs (called sunk costs), project forking, lack of internal expertise, hard selection process (reported as knowledge barriers), legacy integration, and technological immaturity. In turn, Rossi et al. (2012) investigated two Italian governmental organizations as case studies. These investigators found the following main enablers: positive attitude to change (akin new business opportunities in private sector), top management support, employee skill and training, and implementation champion. Among the inhibitors these ones were found: legacy integration, lack of training, and technology complexity. Finally, in Li et al.'s study (2013) were surveyed 104 FLOSS-adopting organizations and 111 non-adopting organizations in

China. They found the following factors in adopting organizations: small IT department size, availability of internal support, availability of external support, and moderated IT criticality.

Table 3 and 4 summarizes the set of main enablers and inhibitors found in these 15 studies. In Tables 3 and 4 we have linked some attributes (like functionality-quality; security-reliability, among others) for keeping a manageable data table. We have also renamed some attributes for grouping in a single one with similar meaning (like legacy integration for several specific attributes reported as interconnectedness, interoperability, and others one). From Table 3, we identify the most recurrent enablers for implementing successfully FLOSS tools: functionality-quality (9 times), no license costs (7 times), avoidance of proprietary lock-in (5 times), and customizability, security-reliability and top management support (4 times). From Table 4 identify the most reported inhibitors impeding a successful FLOSS tool implementation: lack of internal support (7 times), lack of external support (5 times), and lack of supplier responsibility and switching costs (4 times).

Table 3. Common acceptance FLOSS enablers

FLOSS Research	**Avoidance of proprietary**	**Community support**	**Compatibility**	**Customizability**	**Ease of use**	**Employee skill and training**	**Functionality – quality**	**Implementation champion**	**New business opportunities**	**No license costs**	**Performance - efficiency**	**Standard compliance**	**Small IT size**	**Security - reliability**	**User involvement**	**Top management support**	**Trialability**
Dedrick and West (2003)	~	~	●	~	●	~	●	~	●	●	~	~	~	●	~	~	~
Fitzgerald and Kenny (2004)	●	~	~	~	~	~	~	~	~	●	~	~	~	~	●	●	~
Goode (2005)	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~	~
Holck et al. (2005)	~	~	~	~	~	~	●	~	~	●	~	~	~	~	~	~	~
Waring and Maddocks (2005)	~	~	~	●	~	~	●	~	~	~	~	~	~	●	~	~	~
Verma et al. (2005)	~	~	●	~	●	~	~	~	~	~	~	~	~	~	~	~	~
Ven and Verelst (2007)	●	~	~	~	~	●	●	~	~	●	~	●	~	~	~	~	●
Morgan and Finnegan (2007)	●	~	●	●	~	~	●	●	●	●	●	●	~	●	~	●	~
Gallego et al. (2008)	~	~	~	●	●	~	●	~	~	~	~	~	~	~	~	~	~
Sohn and Mok (2008)	~	●	~	●	~	~	●	~	~	~	~	~	~	●	~	~	~
Lee et al. (2009)	~	●	~	~	~	~	●	~	~	~	~	~	~	~	~	~	~
Hauge et al. (2010)	●	●	~	~	~	●	●	~	~	●	~	~	~	~	~	●	●
Nagy et al. (2010)	●	~	~	~	~	~	~	~	~	●	~	~	~	~	~	~	~
Rossi et al. (2012)	~	~	~	~	~	●	~	●	●	~	~	~	~	~	~	●	~
Li et al. (2013)	~	~	~	~	~	~	~	~	~	~	~	~	●	~	~	~	~
Frequency	**5**	**3**	**3**	**4**	**3**	**3**	**9**	**2**	**3**	**7**	**1**	**2**	**1**	**4**	**1**	**4**	**2**

Table 4. Common acceptance FLOSS inhibitors

FLOSS Framework	**Hard selection process**	**IT staff resistance**	**Lack of documentation**	**Lack of external support**	**Lack of internal expertise**	**Lack of supplier**	**Legacy integration**	**Project forking**	**Switching costs (training)**	**Technological complexity**	**Technological immaturity**	**Uncontrolled utilization**
Dedrick and West (2003)	~	~	~	●	●	~	~	~	~	~	~	~
Fitzgerald and Kenny (2004)	~	●	~	~	~	●	~	~	~	~	~	~
Goode (2005)	~	●	~	~	~	●	~	~	~	~	~	~
Holck et al. (2005)	~	~	~	●	●	●	~	~	●	~	~	●
Waring and Maddocks (2005)	~	~	~	~	~	~	~	~	~	~	~	~
Verma et al. (2005)	~	~	~	~	~	~	~	~	~	~	~	~
Ven & Verelst (2007)	~	~	~	●	●	~	~	~	●	~	~	~
Morgan and Finnegan (2007)	●	~	●	●	●	~	~	~	~	~	~	~
Gallego et al. (2008)	~	~	~	~	~	~	~	~	~	~	~	~
Sohn and Mok (2008)	~	~	~	~	~	~	~	~	~	~	~	~
Lee et al. (2009)	~	~	~	~	~	~	~	~	~	~	~	~
Hauge et al. (2010)	●	~	~	●	●	●	~	~	●	~	~	~
Nagy et al. (2010)	●	~	~	~	●	~	●	●	●	~	●	~
Rossi et al. (2012)	~	~	~	~	~	~	●	~	~	●	~	~
Li et al. (2013)	~	~	~	~	●	~	~	~	~	~	~	~
Frequency	**3**	**2**	**1**	**5**	**7**	**4**	**2**	**1**	**4**	**1**	**1**	**1**

Finally, from the Tables 2, 3 and 4, we have derived a final integrative set of 32 attributes. These are reported in Tables 5 and 6. These attributes are the most frequent and shared attributes collected from both core literatures: FLOSS evaluation models and FLOSS adoption models. These Tables 5 and 6 report the attribute, its definition, and its associated scale of 5 levels of risk (certain, high, moderate, low or null).

Table 5. Set core of FLOSS attributes (part I)

Attribute	**Definition**	**Null risk value**	**Low risk value**	**Moderate risk value**	**High risk value**	**Certain risk value**
Community support	Availability of technical support for tool utilization.	Very high-low cost	High-low cost	Sufficient-high cost	Scarce-high cost	None-high cost
Cost effectiveness	Financial impact of utilization of the tool.	Very high	High	Moderate	Low	Very low
Customizability	Extent of tailoring of tool for specific	Very high	High	Moderate	Low	Very low

	requirements.					
Developer community	The expertise level of the tool development teams.	Very high	High	Moderate	Low	Very low
Developer org. structure	Mode of operation and management of the development teams.	Totally organized	Organized	Fairly organized	Disorganized	Totally disorganized
Development process	Quality of the tool development process.	Very high	High	Moderate	Low	Very low
Documentation	Availability of technical and user manuals and extra documents.	Very high	High	Moderate	Low	Very low
External reviews	Availability of third-party technical reviews of the tool.	Very high	High	Moderate	Low	Very low
Functionality - quality	Extent of expected and enhanced functionalities provided by the tool.	Very high	High	Moderate	Low	Very low
Interested IT staff	Extent of willingness of use and interest of IT staff on the new FLOSS tool.	Very high	High	Moderate	Low	Very low
Internal expertise	Existence of FLOSS expertise in the organization.	Very high	High	Moderate	Low	Very low
Interoperability - portability	Extent of intercommunication with other tools or running in several platforms.	Very high	High	Moderate	Low	Very low
Licensing	Characteristics of tool licenses.	Null restrictions	Minimal restrictions	Partial restrictions	Critical restrictions	Very critical restrictions
Maintainability	Extent of continued corrective and improved maintenance of tool.	Very high	High	Moderate	Low	Very low
Market image - popularity	Reputation of the tool.	Very positive	Positive	Fairly positive	Contrasted	Very contrasted
Maturity - longevity	Period of first release of tool.	Decades	Several years	One year	Few months	One month
New business opportunity	Extent of introducing an innovative business process supported by the tool.	Very high	High	Moderate	Low	Very low

Table 6. Set core of FLOSS attributes (part II)

Attribute	**Definition**	**Null risk value**	**Low risk value**	**Moderate risk value**	**High risk value**	**Certain risk value**
Performance - efficiency	Extent of adequate response-times for any operation in the tool.	Very high	High	Moderate	Contrasted	Very contrasted
Project champion	Existence of a respected member of the organization fostering the FLOSS tool.	Several ones	Two	One	Partial support	None
Project forking	Extent of multiple versions and developer teams for the tool.	Very low	Low	Moderate	High	Very high
Security - reliability	Extent of error-free status and hidden-flaws of the tool.	Very low	Low	Moderate	High	Very high
Skilled user group	Existence of well-trained user groups for the new FLOSS tool.	Very high	High	Moderate	Low	Very low
Standards compliance	Compliance of tool with current and domain-related standards.	Very high	High	Moderate	Low	Very low
Switching costs	Extent of overall costs caused for the FLOSS adoption.	Very low	Low	Moderate	High	Very high
Technical environment	Quality and maturity of the technical environment used for the tool.	Very high	High	Moderate	Low	Very low
Test information	Availability of tool test reports.	Very high	High	Moderate	Low	Very low
Top management support	Extent of the economic and political support from highest level management.	Very high	High	Moderate	Low	Very low
Training	Availability of free or affordable user and technical courses.	Very high	High	Moderate	Low	Very low
Usability	Easiness of installation, learning and utilization of the tool.	Very high	High	Moderate	Low	Very low
Usefulness - relevance	Extent of advantage relative perceived by users on the FLOSS tool.	Very high	High	Moderate	Low	Very low
User community	Scope and size of current active community of tool users.	Global	Continental	International	National	Regional
User involvement	Extent of user participation for FLOSS implementation in the organization.	Very high	High	Moderate	Low	Very low

3 The design of the MADM risk-based model for evaluation-selection of FLOSS tools

3.1 Foundations on MADM

A MADM mechanism is a procedure for making preference decisions (e.g. evaluating, prioritizing, and selecting) over a set of available courses of action, where each one is associated usually conflictive levels of attributes (Yoon and Hwang, 1995; Roy, 2005). A MADM risk-based approach can be defined as a decision-making mechanism based on conflictive attributes whose assessment of the courses of action are based on the levels of risk exposition. A risk exposition can be defined as the net expected damage on an asset of interest exercised on an asset's vulnerability by considering jointly the likelihood of occurrence and the impact. This joint consideration is combined usually by a qualitative scale of low, moderate and high-risk exposition (Stoneburner et al., 2002).

3.2 The iterative design process of the MADM risk-based evaluation-selection FLOSS tool model

Based on FLOSS adoption frameworks and FLOSS adoption models literature, we pose the convergence of both ones through a risk-based approach (Stoneburner et al., 2002). For fostering a practical utilization of it, we pose to generate a FLOSS success implementation value tree, which can be operationalized through a MADM model. With this MADM model, ITSM practitioners interested in evaluating two or more FLOSS alternatives will be able to assess the overall estimated success implementation value of each alternative by evaluating the risk-based attributes.

A decision value tree structure is a hierarchy of compose of an overall expected objective (highest level), a set of related preferred sub-objectives (intermediate level), and a set of related attributes (lowest level) used as the measurement dimension against each course of action (e.g. an alternative action) will be assessed in a decision-making process. Attributes are also known as performance measures, figures of merit, metrics or criteria. Consequently, the set of courses of action (alternatives of action) are not included in a decision value tree structure.

A decision value tree structure is based on the own concerns and preferences of the decision-maker or decisional group. However, decision-making literature (Buede, 1986; Huber and McDaniel, 1986; Simon, 1997) suggests that a well-structured and informed decision-making process from intelligence, design, choice, to implementation and learning phases, can generate better benefits that an informal process. Thus, for generating the decision value tree, three general approaches have been suggested (Buede, 1986):

1. to conduct a literature review on relevant studies associated to the problem;
2. to elaborate an input-process-output model and derive relevant objectives and metrics;
3. to interview experts and elaborate an ad-hoc value scheme.

In this research we use the first approach and we design a decision value tree from the FLOSS adoption frameworks and FLOSS adoption models literature. This decision value tree is elaborated from the set of factors whose existence or lack of have been associated with successful implementations of FLOSS tools, and thus can be considered an informed process (idem, 1986). Buede (1986) suggests that either a top-down (from highest level objectives to sub-objectives and lowest level attributes path) or a bottom-up (from lowest level attributes to highest level sub-objectives and objectives path) approach can be specifically followed for structuring the value tree. In this research we combine these two approaches, and we follow additional recommendations from literature (Buede, 1986; Keeney and Gregory, 2005).

The steps conducted from this hybrid approach were as follows:

1. to state the overall top objective (top-down approach);
2. to identify sub-objectives from the top objective if required, and to repeat this step if required for each sub-objective (top-down approach);
3. to iteratively complete the full hierarchy of attributes (also called criteria) by using the initial list of attributes in the lowest level of the hierarchy, which will be logically associated to the set of previous identified sub-objectives (lowest level) (top-down and bottom-up approaches);
4. to refine the initial list of lowest level attributes based on the literature recommendations on unambiguity, comprehensiveness, directionality, operationability and understandability (Keeney and Gregory, 2005);
5. to assess the value tree hierarchy on completeness, operationability, decomposability, lack of redundancy, and size (Buede, 1986).

For Buede (1986) (based on Keeney and Raiffa (1976)), a decision value tree must be complete, operational, decomposable, non-redundant, and minimal. A decision value tree is complete when it covers all concerns for decision makers. It is operational when the courses of action being considered can be clearly characterized. It is decomposable when there is independence between preferences and uncertainties, so an objective can be divided in sub-objectives. Non redundant implies that there are not overlapping objectives, sub-objectives or attributes, and it avoids double counting of influences (Buede, 1986). Finally minimal implies that once satisfied the previous conditions the value structure should not be unnecessary incremented.

In particular Keeney and Gregory (2005) have proposed that adequate attributes in the lowest level of the value tree hierarchy should achieve the following properties: unambiguity, comprehensiveness, directionality, operationability and understandability. Unambiguity means the attribute measures clearly and uniquely the level of consequences on the related objectives (e.g. fitness to objectives). Comprehensiveness implies that the attribute covers sufficiently the conceptual dimensionality of consequences on the related objectives (e.g. adequacy to objectives). Directionality means that the attribute scale of value describes directly the consequences on related objectives (e.g. measurement of objectives). Operationability implies that the attribute can be measured by reasonable and reachable information (e.g. affordability to objectives). Understandability means that the consequences assessed for objectives by using the attribute are readily understood and communicated.

3.3 Application of the iterative design process

Step 1. To state the overall top objective (top-down approach). We (research team) are interested in elaborating a decisional value tree hierarchy useful for choosing the FLOSS tool with the minimum overall implementation risk for a particular small-medium sized organization. Thus, the overall top objective is stated as: BEST (MINIMUM OVERALL IMPLEMENTATION RISK) FLOSS TOOL.

Step 2. To identify sub-objectives from the top objective if required, and to repeat this step if required for each sub-objective (top-down approach). The set of sub-objectives identified in this step (ii) correspond to the categories of: FINANCIAL RISKS, ORGANIZATIONAL

RISKS, END-USER RISKS, and TECHNICAL RISKS. For this initial research, we consider sufficient to limit to a single division of the overall goal. This group of sub-objectives can be also divided, but it is recommend to elaborate a parsimoniously value structure.

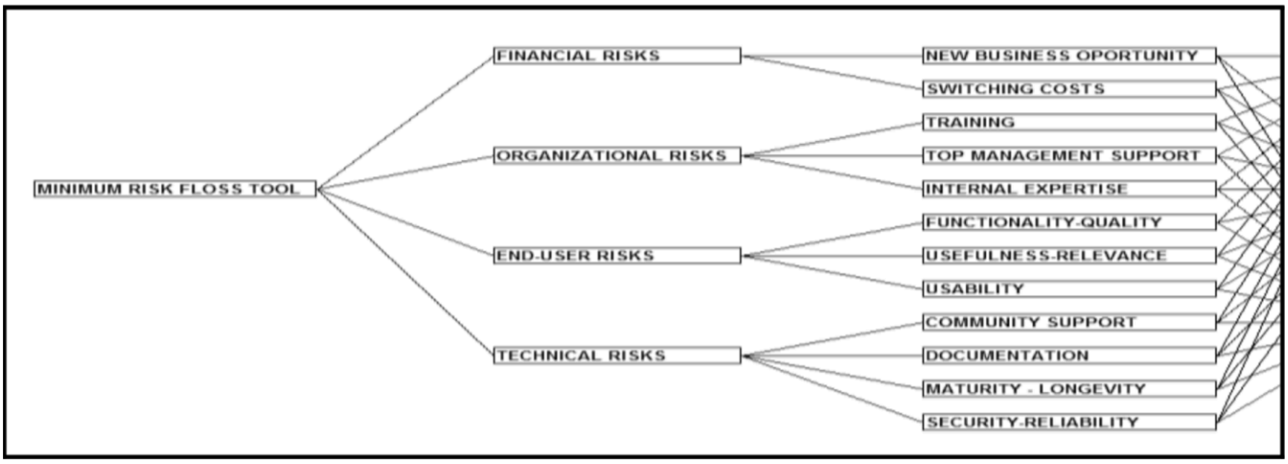
Step 3. To iteratively complete the full hierarchy of attributes (also called criteria) by using the initial list of attributes in the lowest level of the hierarchy, which will be logically associated to the set of previous identified sub-objectives (lowest level) (top-down and bottom-up approaches). From the two FLOSS evaluation frameworks and FLOSS adoption models literatures we identified 32 attributes. These attributes, in this risk-based evaluation approach, are considered risky events with a high, moderate or low risk exposure determined by the joint consideration of their likelihood of occurrence and their negative impact on financial, organizational, end-user and technical goals of the organization. It must be noted that for each specific FLOSS tool alternative to be evaluated, each attribute will generate a specific risk exposure value. We elaborate an initial full hierarchy by assigning each attribute to its most adequate category of risk among FINANCIAL, ORGANIZATIONAL, END-USER or TECHNICAL one. This process was conducted several times by research team, until a final agreement. Few discrepancies were found, and those were agreed in next iteration. The assignation of the 32 attributes to the 4 risk categories was as follows:

- **FINANCIAL RISKS**: licensing cost effectiveness, new business opportunity, and switching costs.
- **ORGANIZATIONAL RISKS**: external reviews, internal expertise, interested IT staff, project champion, skilled end-user group, top management support, training, usability, and user involvement.
- **END-USER RISKS**: functionality-quality, market image, performance-efficiency, and usefulness-relevance.
- **TECHNICAL RISKS**: community support, development process, developer community, developer org. structure, documentation, interoperability-portability, maintainability, maturity-longevity, project forking, security-reliability, test information, standard compliance, technical environment, and user community.

Step 4. To refine the initial list of lowest level attributes based on the literature recommendations on unambiguity, comprehensiveness, directionality, operationability and understandability (Keeney and Gregory, 2005). The Tables 1 to 4 (in the appendix) report the results of the refinement procedure applied to the initial list of 32 attributes. Based on this analysis, a final set of 12 selected attributes for the essential model was identified.

Step 5. To assess the value tree hierarchy on completeness, operationability, decomposability, lack of redundancy, and size (Buede, 1986). In this analysis, we consider the full value tree hierarchy derived from the previous step 4) as showed in Figure 1. The expected completeness, operationability, decomposability, lack of redundancy and size properties were assessed as satisfied. The Table 5 in the appendix reports this analysis on the overall adequacy of the value tree hierarchy.

Figure 1 MADM risk-based evaluation-selection FLOSS tool model



4 Application of the MADM risk-based evaluation-selection FLOSS model

4.1 Relevance of FLOSS tools in the domain of IT service management process

Large and medium sized organizations implement Information Technology Service Management (ITSM) Process Frameworks (mainly ITIL v2, ITIL v3, ISO/IEC 20000 or MOF 4.0) with the aim to provide organizational value through the delivery of IT services under a cost-effective management of IT capabilities and IT resources (Gallup et al., 2009). However, the implementation and finally operation of an ITSM Process Framework demands the investment of financial, human and other organizational resources. In particular, the utilization of software tools is suggested for coping with the inherently complexity of the ITSM process administration (caused by the required utilization of multiple processes, interrelationships and data) (Brenner, 2006). However, while large and medium sized organizations can afford commercial tools from a wide offering, the involved costs preclude it for small organizations. Thus, the availability of FLOSS tools becomes a potential feasible alternative for small organizations.

4.2 Illustrative demo case

This value hierarchy was implemented in MADM mechanism by using an academic version of the Criterium Decision Plus tool (CDP, 2015). Three ITSM FLOSS tools (ITOP, IDOIT and OTRS) were evaluated as FLOSS tools for supporting the configuration management ITSM process. The Table7 reports the input data assessed for each one of the 12 attributes in the MADM FLOSS evaluation-selection model. The Table 8 reports the transformed risk-based input data. In this MADM model we assume a similar level of importance for the four categories of risks (i.e. FINANCIAL, ORGANIZATIONAL, END-USER and TECHNICAL). Thus their level of importance was 0.250 for each category. We assume also a similar equalized scheme for all sub-criteria in each category. For example, in the ORGANIZATIONAL risk category there are three sub-criteria, and thus each one had an importance weight of 0.333.

Table 7. Input data for IDOIT, ITOP and OTRS FLOSS tools

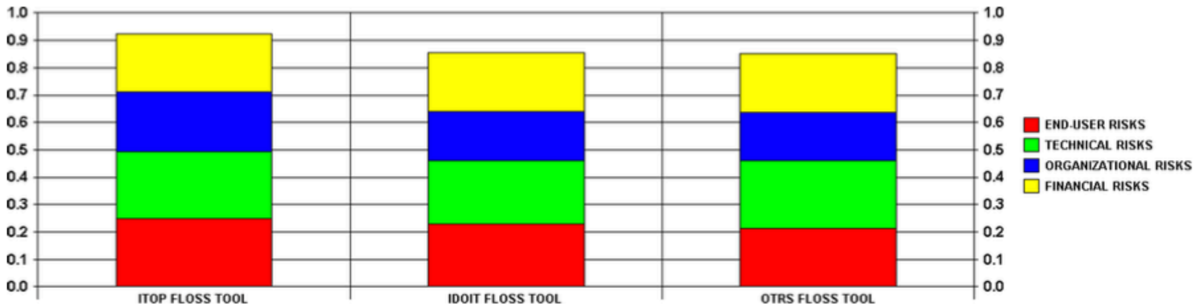
Risk Attribute	Definition	Risk Category	IDOIT	ITOP	OTRS
New business opportunity	Extent of introducing an innovative business process supported by the tool.	Financial	Moderate	Moderate	Moderate
Switching costs	Extent of overall costs caused for the FLOSS adoption.	Financial	Low	Low	Low
Training	Availability of free or affordable user and technical courses.	Organizational	Low	High	High
Top management support	Extent of the economic and political support from highest level management.	Organizational	Moderate	Moderate	Moderate
Internal expertise	Existence of FLOSS expertise in the organization.	Organizational	High	High	Low
Functionality - quality	Extent of expected and enhanced functionalities provided by the tool.	End user	High	Very high	High
Usefulness - relevance	Extent of advantage relative perceived by users on the FLOSS tool.	End user	High	Very high	High
Usability	Easiness of installation, learning and utilization of the tool.	End user	Moderate	High	Low
Community support	Availability of technical support for tool utilization.	Technical	Scarce-high cost	High-low cost	High-low cost
Documentation	Availability of technical and user manuals and extra documents.	Technical	Moderate	High	High
Maturity – longevity	Period of first release of tool.	Technical	Several years	Several years	Several years
Security - reliability	Extent of error-free status and hidden-flaws of the tool.	Technical	Moderate	Moderate	Moderate

Table 8. Transformed risk-based input data for IDOIT, ITOP and OTRS FLOSS tools

Risk Attribute	Risk Category	IDOIT	ITOP	OTRS
New business opportunity	Financial	Moderate risk	Moderate risk	Moderate risk
Switching costs	Financial	Low risk	Low risk	Low risk
Training	Organizational	High risk	Low risk	Low risk
Top management support	Organizational	Moderate risk	Moderate risk	Moderate risk
Internal expertise	Organizational	Low risk	Low risk	High risk
Functionality - quality	End user	Low risk	Null risk	Low risk
Usefulness - relevance	End user	Low risk	Null risk	Low risk
Usability	End user	Moderate risk	Low risk	High risk
Community support	Technical	High risk	Low risk	Low risk
Documentation	Technical	Moderate risk	Low risk	Low risk
Maturity – longevity	Technical	Low risk	Low risk	Low risk
Security - reliability	Technical	Moderate risk	Moderate risk	Moderate risk

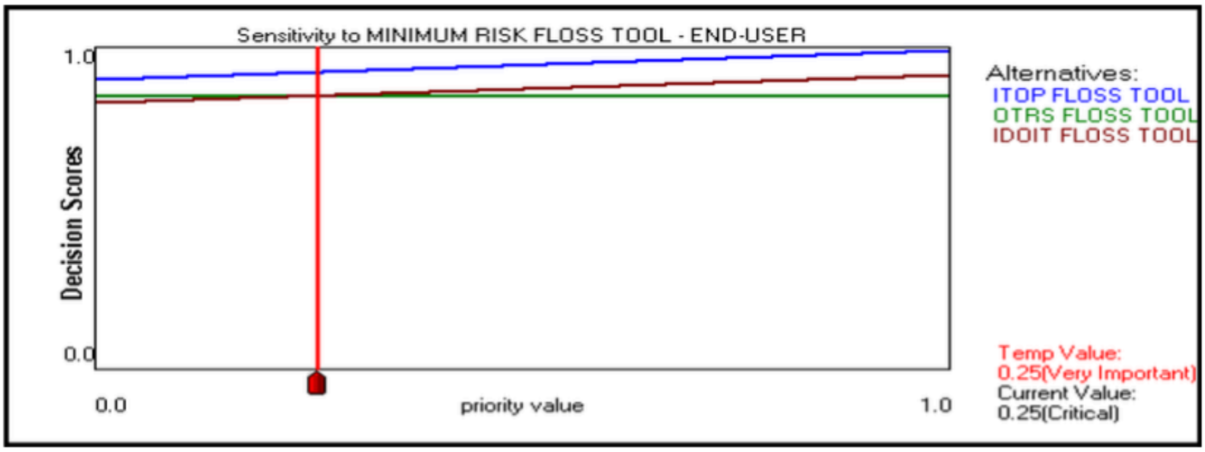
The figures from 2 to 4 report the results of this demonstrative evaluation of the three FLOSS ITSM tools. The figure 2 shows the overall risk-based scores obtained by the three FLOSS tools. These scores are in the range from 0.000 to 1.000. Lower scores are associated to lower level of risk. The MADM scores obtained by IDOIT, OTRS and ITOP tools were respectively 0.854, 0.924, and 0.851. Thus, the ITOP FLOSS tool is rated as the best tool in this evaluation. The figure 2 shows also the contributions to this overall risk-based score from the four criteria (END-USER, TECHNICAL, ORGANIZATIONAL and END-USER ones reported from bottom to top). The ITOP FLOSS tool was rated with less TECHNICAL, ORGANIZATIONAL and END-USER level of risk, despite a similar level risk in FINANCIAL type regarding the other two FLOSS tools, and similar level of risk in TECHNICAL dimension regarding to the OTRS tool. Thus, ITOP tool outperforms (with less overall risk) to other two tools (IDOIT and ITOP) regarding the ORGANIZATIONAL and END-USER risks.

Figure 2 Overall evaluation of risk-based score of the three FLOSS ITSM tools



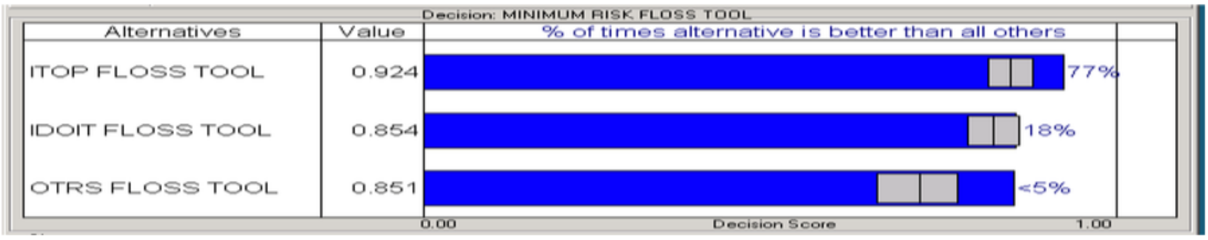
A sensitivity analysis was conducted on changes in the weight scheme for the ORGANIZATIONAL and END-USER categories of risks. In both cases, none change in the importance weight scheme modified the end result of ITOP tool as best one (i.e. overall highest value by least overall risk). Figure 3 shows it for the END-USER risk category.

Figure 3 Sensitivity analysis on the END-USER risk category



The Figure 4 reports an uncertainty analysis. This analysis consists in assigning a probability distribution function for each input value assigned to each one of the 12 sub-criteria for the three evaluated FLOSS tools. In this illustrative case, we use a normal distribution with media similar to the initial assessed value for each sub-criterion, and a standard deviation of 0.10. For example, the initial numerical input value for the sub-criteria *Training* for the IDOIT, ITOP and OTRS tools were respectively 0.75, 0.25 and 0.25, which corresponded to high risk for IDOIT and low risk for ITOP and OTRS. These values are actually assigned qualitatively for the decision-maker from a scale to null, low, moderate, high and certain risk level. These five qualitative values correspond to the weights 0.00, 0.25, 0.50, 0.75 and 1.00 respectively. The uncertainty probability distribution functions assigned were normal distributions with means of 0.75, 0.25, and 0.25 respectively and a standard deviation of 0.10 for the three FLOSS tools. The overall integrated results indicated that the ITOP tool can be considered as best one (i.e. with an overall minimum risk) from a practical significance. The ITOP FLOSS tool outperformed a 77% to other two tools under a Sensitivity Analysis of uncertainty on the scores assigned to the 12 attributes for each FLOSS tool. However, from a statistical significance we cannot reject the hypothesis on that ITOP is better than IDOIT because this last one FLOSS tool outperformed 18% to ITOP and OTRS (i.e. a percentage greater than 10%). This last statistical-based result must be interpreted with caution as it relies totally from the statistical distribution assigned and assumed for each one of the scores assigned to the 12 criteria. In summary, the recommendation for this illustrative case is to select and implement ITOP FLOSS tool with cautionary statistical consideration on IDOIT FLOSS tool.

Figure 4 Uncertainty analysis on all risk-based input data



5 Conclusions

The phenomenon of elaboration and potential acceptance of FLOSS has permeated worldwide large organizations. This has been generated by:

1. the acknowledgement of successful FLOSS systems such as: Linux operating system, Apache web server, MySQL data base management system, OpenOffice suite, among others;
2. the endorsement for some FLOSS products from large IT companies;
3. the openness to access and modify source code;

the free-cost license scheme. However, as it was indicated by the reviewed literature, to select and implement the correct FLOSS tool is not a straightforward process and a wrong selection can lead to a loss of valuable organizational resources. Furthermore, when the FLOSS acceptance and implementation process is pursued for small and medium sized organizations, which should be natural by the free-cost license scheme, additional complications emerge by the complexity of some free-access or proprietary FLOSS evaluation models or the consulting costs for elaborating a suitable FLOSS evaluation model for a specific small or medium sized organization.

Thus, in this research we reviewed the FLOSS evaluation-selection framework and FLOSS success-failure implementation factor literatures to advance on such studies through the design of an essential FLOSS evaluation-selection model which emerged from both literatures. Our FLOSS evaluation-selection model was designed with a risk-based decision making approach. This model was structured as a Multi-Attribute Decision Making (MADM) model which included 12 attributes grouped in four risk categories: financial, organizational, end-user and technical ones. We illustrated also its utilization in the domain of Information

Technology Service Management (ITSM) FLOSS tools. For it, we utilized a demo version of the Criterium DecisionPlus MADM tool. We consider relevant also to establish the following three statements as cautionary limitations:

1. while we identified initially 32 core attributes from two main reviewed literatures, the final model of 12 attributes organized in four categories was totally dependently for the interpretation on ambiguity, operationability, understandability, and relevance done by the research team;
2. the correctness of evaluation-selection process is highly dependently on the correctness of the scores assigned to each one of the 12 attributes assigned to the evaluated FLOSS tools;
3. the correctness of results from the Sensitivity Analysis of uncertainty are also highly from the extent of correctness on the assumed probability distributions and assigned parameters to each one of the 12 attributes assigned to the evaluated FLOSS tools.

Finally, we recommend advancing this research through of the following paths:

1. to conduct empirical research with small and medium sized organizations to assess the usability (i.e. usefulness, ease of use, compatibility, and value) of the FLOSS evaluation-selection model;
2. to conduct empirical research with small and medium sized organizations on the refinement process to elaborate an essential FLOSS evaluation-model from the initial set of 32 attributes identified from core two literatures;
3. to conduct experimental research for assessing the comparative usability of the proposed new FLOSS evaluation model versus other FLOSS evaluation model reported in the literature. Hence, our model contributes to the FLOSS evaluation-selection literature with the inclusion of the risk management approach and to the FLOSS evaluation-selection praxis with the provision of an essential evaluation-selection model for FLOSS tools for small organizations derived from two core set of FLOSS literature.

Acknowledgments

This research was supported by the PIINF14-3 grant provided by the Autonomous University of Aguascalientes for the period 2014-2015. Authors thank also our affiliated institutions for providing additional resources.

References

Androutsellis-Theotokis, S., Spinellis, D., Kechagia, M. and Gousios, G. (2011) 'Open source software: A survey from 10,000 feet', *Foundations and Trends in Technology, Information and Operations Management*, Vol. 4, No.3-4, pp. 187-347.

Aversano, L. and Tortorella, M. (2013) 'Quality evaluation of floss projects: Application to ERP systems', *Information and Software Technology*, Vol. 55, No. 7, pp. 1260-1276.

Brenner, M. (2006) 'Classifying ITIL Processes; A Taxonomy under Tool Support Aspects', in IEEE/IFIP International workshop on Business-Driven IT Management, Vancouver, Canada, pp. 19-28.

Buede, D. (1986) 'Structuring Value Attributes', *Interfaces*, Vol. 16, No. 2, pp. 52-62.

CDP (2015). *Criterium Decision Plus software academic demo tool*, available online at: http://www.infoharvest.com/ihroot/infoharv/cdpfreedownloads.asp (accessed on March 15, 2015).

Coar, K. (2005) *The open source definition*, available online at: http://opensource.org/osd (accessed on March 15, 2015).

David, P., Waterman, A. and Arora, S. (2003) *FLOSS-US The Free/Libre Open Source Software Survey for 2003*, available online at: http://web.stanford.edu/group/floss-us/ (accessed on March 15, 2015).

Dedrick, J. and West, J. (2003) 'Why Firms adopt Open Source Platforms: a Grounded Theory of Innovation and Standards Adoption', in Workshop on standard making: A critical research frontier for information systems, Seattle, USA, pp. 236-257.

del Bianco, V., Lavazza, L., Morasca, S. and Taibi, D. (2009) 'Quality of open source software: The QualiPSo trustworthiness model', in 5th IFIP WG 2.13 International Conference on Open Source Systems, Skövde, Sweden, pp. 199-212.

Depietro, R., Wiarda, E. and Fleischer, M. (1990) 'The Context for Change: Organization, Technology and Environment', in Tornatzky, L. and Fleischer, M. (Eds.): *The processes of technological innovation*, Lexington Books, Lexington, MA, USA, pp. 151-175.

Deprez, J. (2008) *QualOSS Final Activity Report*, available online at: https://www.cetic.be/IMG/pdf/ WP7_QualOSS_Final_Activity_Report_Submitted.pdf (accessed on March 15, 2015).

Duijnhouwer, F. and C. Widdow (2003) 'Capgemini Expert Letter Open Source Maturity Model'. *Capgemini Expert Letter*, August issue, pp. 1-18,.

Fitzgerald, B. and Kenny, T. (2004) 'Developing an information systems infrastructure with open source software', *IEEE Software*, Vol. 21, No. 1 pp. 50-55.

Gallego, M., Luna, P. and Bueno, S. (2008) 'User acceptance model of open source software', *Computers in Human Behavior*, Vol. 24, pp. 2199-2216.

Gallup, S., Dattero, R., Quan, J. and Conger, S. (2009) 'An Overview of IT Service Management', *Communications of the ACM*, Vol. 52, No. 5, pp. 124-127.

Golden, B. (2005) *Creating and Implementing an Open Source Policy: Five Steps to Success,* available online at: http://carahsoft.biz/pdf/NavicaOpenSourcePolicyProcess Whitepaper.pdf (accessed on March 15, 2015).

Goode, S. (2005) 'Something for nothing: management rejection of open source software in Australia's top firms', *Information & Management*, Vol. 42, No. 5, pp. 669-681.

Hauge, Ø., Cruzes, D., Conradi, R., Velle, K. and Skarpenes, T. (2010) 'Risks and Risk Mitigation in Open Source Software Adoption: Bridging the Gap between Literature and Practice', in 6th IFIP WG 2.13 International Conference on Open Source Systems, Notre Dame, USA, pp. 105-118.

Holck, J., Michael, M., Larsen, H. and Pedersen, M. (2005) 'Managerial and technical barriers to the adoption of open source software', in 4th International Conference ICCBSS, Bilbao, Spain, pp. 289-300.

Huber, G. and McDaniel, R. (1986) 'The Decision-Making Paradigm of Organizational Design', *Management Science*, Vol. 32, No. 5, pp. 572-589.

ISO (1991) *ISO/IEC 9126 Information technology–Software product evaluation–Quality characteristics and guidelines for their use*, ISO/IEC, Geneva, Switzerland.

ISO (2005) *ISO 9000 Quality Management Systems--Fundamentals and Vocabulary*, ISO/IEC, Geneva, Switzerland.

ISO (2011) *ISO/IEC 25010 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models*, ISO/IEC, Geneva, Switzerland.

Keeney, R. and Gregory, R. (2005) 'Selecting Attributes to Measure the Achievement of Objectives', *Operations Research*, Vol. 53, No. 1, pp. 1-11.

Keeney, R., Ralph, L. and Raiffa, H. (1976) *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, John Wiley and Sons, New York, NY, USA.

Lee, S., Kim, H. and Gupta, S. (2009) 'Measuring open source software success', *Omega*, Vol. 37, pp. 426-438.

Li, Y., Tan, C. and Yang, X. (2013) 'It is all about what we have: A discriminant analysis of organizations' decision to adopt open source software', *Decision Support Systems*, Vol. 56, pp. 56-62.

Mora, M., Gomez, J., O'Connor, R. V. and Gelman, O. (2015), 'A Risk-based MADM Model for Selecting Free-Libre Open Source Software Tools in the Domain of IT Service Management', in 16th. Symposium on Decision Technology and Intelligent Information Systems, Baden-Baden, Germany, pp.

Morgan, L. and Finnegan, P. (2007) 'How perceptions of open source software influence adoption: an exploratory study', in 15th. European Conference on Information Systems, St. Gallen, Switzerland, pp. 973-984.

Nagy, D., Yassin, A. and Bhattacherjee, A. (2010) 'Organizational adoption of open source software: barriers and remedies', *Communications of the ACM*, Vol. 53, No.3, pp. 148–151.

OpenBRR.org (2005) *Business Readiness Rating for Open Source - A Proposed Open Standard to Facilitate Assessment and Adoption of Open Source Software*, available online at: http://www.openbrr.org (accessed on March 15, 2015).

Petrinja, E., Nambakam, R. and Sillitti, A. (2009) 'Introducing the opensource maturity model'. in ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development, Washington, USA, pp. 37-41.

QSOS.org (2013) *Qualification and Selection of Open Source software v2 (QSOSv2)*, available online at: http://www.qsos.org/Method.html (accessed on March 15, 2015).

Rossi, B., Russo, B. and Succi, G. (2012) 'Adoption of free/libre open source software in public organizations: factors of impact', *Information Technology & People*, Vol. 25, No. 2, pp. 156-187.

Roy, B. (2005) 'An Overview of MCDA Techniques Today', in Figueira, J., Greco, S., and Ehrgott, M. (Eds.): *Multiple Criteria Decision Analysis–State of the Art Surveys*, Springer, Heidelberg, Germany, pp. 3-26.

Samoladas, I., Gousios, G., Spinellis, D. and Stamelos, I. (2008) 'The SQO-OSS quality model: measurement based open source software evaluation', in Russo, B., Damiani, E., Hissam, S., Lundell, B. and Succi, G. (Eds.): *Open source development, communities and quality*, Springer, Boston, MA, USA, pp. 237-248.

Simon, H. (1997) *Administrative Behavior: a study of Decision-Making process in Administrative Organizations*, Free Press, New York, NY, USA.

Sohn, S. and Mok, M. (2008) 'A strategic analysis for successful open source software utilization based on a structural equation model', *The Journal of Systems and Software*, Vol. 81, pp. 1014–1024.

Spinellis, D. and Giannikasa, V. (2012) 'Organizational Adoption of Open Source Software', *Journal of Systems and Software*, Vol. 85, No. 3, pp. 666–682.

Stol, K. J. and Babar, M. A. (2010) 'A comparison framework for open source software evaluation methods', in Ågerfalk, P., Boldyreff, C., González-Barahona, J., Madey, G. and Noll, J. (Eds.): *Open Source Software: New Horizons,* Springer, Heidelberg, Germany, pp. 389-394.

Stoneburner, G., Goguen, A. and Feringa, A. (2002) *Risk Management Guide for Information Technology Systems*, National Institute of Standards and Technology, Gaithersburg, MD, USA.

Sung, W., Won, J., Kim, J. and Rhew, S. (2007) 'A quality model for open source software selection', in 6th. Int. Conf. in Advanced Language Processing and Web Information Technology, Luoyang, China, pp. 515-519.

Taibi, D., Lavazza, L. and Morasca, S. (2007) 'OpenBQR: a framework for the assessment of OSS," in Feller, J., Fitzgerald, B., Scacchi, W. and Sillitti, A. (Eds.): *Open Source Development, Adoption and Innovation*, Springer, Boston, MA, USA, pp. 173–186.

Tribunella, T. and Baroody, J. (2007) Open Source for Accounting and Enterprise Systems," in St. Amant, K. and Still, B. (Eds.): *Handbook of Research on Open Source Software: Technological, Economic, and Social Perspectives*, IGI, Hershey, PA, USA, pp. 555-569.

Ven, K. and Verelst, J. (2006) 'The organizational adoption of open source server software by Belgian organizations', in IFIP Working Group 2.13 Foundation on Open Source Software, Como, Italy, pp. 111-122.

Ven, K., Verelst, J., and Mannaert, H. (2008) 'Should you adopt open source software?', *Software IEEE*, Vol. 25, No. 3, pp. 54-59.

Verma, S., Jin, L. and Negi, A. (2005) 'Open Source Adoption and Use: A Comparative Study Between Groups in the US and India', in AMCIS 2005 Conference, Omaha, USA, 2005, pp. 960-972.

Waring, T. and Maddocks, P. (2005) 'Open Source Software implementation in the UK public sector: Evidence from the field and implications for the future', *International Journal of Information Management*, Vol. 25, No. 5, pp. 411-428.

Watson, R., Boudreau, M., York, P., Greiner, M. and Wynn, D. (2008) 'The business of open source', *Communications of the ACM*, Vol. 51, No. 3, pp. 41-46.

Wheeler, D. (2011) How to Evaluate Open Source Software / Free Software (OSS/FS) Programs. Available: http:// www.dwheeler.com/oss_fs_eval.html

Yoon, K. and Hwang, C. (1995) *Multiple Attribute Decision Making*, Sage University, Thousand Oaks, CA, USA.