Sponsored through Framework Programme Sixth (Call 5) by

Document Information

Version: 1.1
Date : Jan 21, 10
Pages : 39

Owning Partner:
PEPITe

Author(s):
Vincent Auvray (PEPITe)

Reviewer(s):
CETIC

To:
CONSORTIUM

Purpose of distribution:

The QUALOSS Consortium consists of: CETIC (BE), Facultés Notre Dame de la Paix à Namur (BE), Universidad Rey Juan Carlos (ES), Fraunhofer IESE (DE), ZEA Partners (BE), MERIT (NL), AdaCore (FR), PEPITe (BE)

Printed
on 12/15/09 at 04:59:06 PM

Status:

[ ] Draft
[ ] To be reviewed
[ ] Proposal
[X] Final/Released

Confidentiality:

[X] Public - Intended for public use
[ ] Restricted - Intended for QUALOSS consortium only
[ ] Confidential - Intended for individual partner only

Deliverable ID: **D4.3**

Title:

Inferred Quality Models Report

(A data-driven approach to define risk indicators)

Deliverable: D4.3

Title: Inferred Quality Models Report

Executive Summary:

The aim of the QualOSS project is to provide a methodology and tools to rigorously assess Free *libre* Open Source Software and thus facilitate its acquisition. To guide the assessment, a series of role-based questions have been identified. These questions are answered with the help of metrics and risk indicators.

For some metrics, it has proved difficult to design indicators manually. In this deliverable, we propose a methodology to build indicators when there is no relevant knowledge, but some data is available. The methodology, based on the max-entropy principle, is applied to community metrics introduced in D4.2 with data collected by the FLOSSMetrics project. To overcome some shortcomings of these metrics, we also propose a formulation of the underlying risk assessment objective as a prediction problem. This formulation allows us to define new metrics and indicators that better characterize the risk profile of a FlOSS endeavor.

TABLE OF CONTENTS

1. Introduction

The strategic objective of the QualOSS project is to enhance the competitive position of the European software industry by providing a methodology and tools for improving productivity and the quality of software products. To achieve this objective, QualOSS notes that many organizations have started to integrate Free libre Open Source Software (FlOSS) in their systems. Currently, they acquire FlOSS product components based on ad-hoc approaches. It is therefore the aim of QualOSS to facilitate the acquisition of the most adequate FlOSS based on a rigorous assessment methodology.

The Work Package 4 is the core of the QualOSS project. It builds the complete version of the QualOSS methodology and specifies how to apply it. WP4 is divided into several tasks. Let us mention Tasks 4.1, 4.2, and 4.3.

Task 4.1 proposes a generic FlOSS assessment process that can be applied in various FlOSS acquisition contexts, the QualOSS methodology. Also, it proposes a standard way to apply this methodology, the standard QualOSS assessment method, and explains how this method can be customized to answer more advanced questions of more specific and demanding FlOSS acquisition situations.

The standard QualOSS assessment method identified a series of role-based questions to guide the assessment. Task 4.2 completes the standard QualOSS assessment method by identifying adequate metrics and risk indicators to answer these questions. In a few words, an indicator is a function of one or several metrics that indicates a risk level. Task 4.2 also continues to develop the methods and techniques used during an assessment. In particular, it develops an interpretation guide to help users understand the measures and indicators proposed.

The objective of Task 4.3 is to define risk indicators by applying machine learning techniques. This deliverable describes our contributions towards this objective. Robustness and evolvability of FlOSS projects are key aspects for companies with a business model based on FlOSS. They are usually not worried to use products with no stable releases and they highlight the importance of the surrounding community and the support it may provide. Consequently, several metrics measuring various aspects of a community have been defined, in particular its size and regeneration adequacy and its workload adequacy. Section 2. briefly describes the community metrics used in this deliverable. It has proved difficult to define risk indicators based on community metrics manually. On the other hand, it is possible to compute the metrics for a large number of FlOSS projects by using data collected by the FLOSSMetrics project[1]. Section 3. proposes a data-driven approach to define indicators and applies it to the available data. This application highlights several flaws in the definition of some community metrics. Section 4. outlines a new methodology able to overcome these flaws and discusses a preliminary application. Finally, Section 5. illustrates our results on the following FlOSS projects: findbugs, evolution, evince, nautilus, and httpd 1.3.

2. Community metrics

Deliverable D4.2 presents the community metrics used throughout this deliverable. Let us describe them informally in this section. In the Appendix, we provide precise definitions of these metrics in the form of SQL queries applicable to data gathered by the FLOSSMetrics project. For the sake of our analysis, we classify community metrics into

- *trend* or *slope metrics* that measure the evolution of metrics over time intervals and
- *static metrics* that are naturally defined as some overall measure over the entire life of a project.

[1] See http://flossmetrics.org.

2.1 Trend metrics

Given a time interval length T and an open-source project, we divide the lifespan of the project into a sequence of consecutive time intervals $t_1, .., t_n$ of length T. For each such interval, the following *low-level metrics* are defined

- the number **sra2** of new code committers in the interval,
- the number **sra3** of new non-code committers in the interval,
- the number **sra9** of active code committers in the interval,
- the ratio **iwa1** of the number of commits by the number of committers in the interval,
- the ratio **iwa2** of the number of code commits by the number of code committers in the interval,
- the number **sra4** of new core committers in the interval,
- the number **sra5** of core committers that quit in the interval, and
- the difference **sra6** between sra4 and sra5 in the same interval,

where the *core committers* of an interval are the committers responsible for 80% of the commit activity in the interval.

The evolution of a sequence $m(t_1), ..., m(t_n)$ of low-level metric values is captured by a higher level *slope metric* computed as follows. First, we estimate a linear model $y(t) = w_0 + w_1 t/T$ of the sequence by least square regression. Then, we use its slope w_1 as our high-level trend metric. In a slight abuse of notation and language, we denote a low-level metric and its slope metric by the same name.

In this deliverable, we consider time intervals of 30, 90, and 180 days for the slope metrics sra2, sra3, sra9, iwa1, and iwa2. For the slope metrics sra4, sra5, and sra6, we consider a time interval of one year. These last three metrics are taken over a longer period because the notion of core committer requires one to have contributed commits over a significant period and not just in the last 30, 90 or even 180 days.

2.2 Static metrics

The following static metrics are defined in deliverable D4.2 and used here:

- the average number **sra7** of months where each committer committed,
- the proportion **iwa4** of files maintained by a single committer,
- the ratio **iwa5** of the number of lines of source code by the number of committers active in the last year, and
- the proportion **iwa7** of code files committed to in the last year.

3. A data-driven methodology to define indicators

For each community metric M of an open-source project, let us define a high-level risk indicator $R(M)$. This indicator should measure one aspect of the risk taken by a company engaging in a *full floss collaboration* with the project. In Task 4.2 the QualOSS project has defined the following four color-coded risk levels, in order of decreasing risk: *black*, *red*, *yellow* and *green*.

Defining meaningful indicators is not a trivial problem. Before we describe our methodology, let us stress that indicators should not be trusted blindly. In particular, they should not be considered separately, but rather jointly to form an overall risk picture associated to a FlOSS endeavor.

3.1 Description of the methodology

In the absence of knowledge to guide the design of a risk indicator, it is natural to search for an indicator that preserves as much information about its metric as possible. In information-theoretic terms, this principle translates into the following constraint: an indicator R should maximize the mutual information $I(R; M)$ between itself and its metric M. Suppose that we have a dataset of observed metric values available. If

p denotes the distribution of relative frequencies observed in the dataset, the mutual information is defined by

$$I(R;M)=\sum_{r\in R}\sum_{m\in M} p(r,m)\log_2\frac{p(r,m)}{p(r)\,p(m)}. \quad (1)$$

Moreover, since R is a function of M, one can show that $I(R;M)$ is equal to the entropy $H(R)$ with

$$H(R)=-\sum_{r\in R} p(r)\log_2 p(r). \quad (2)$$

One can show that the entropy of a random variable with a finite number k of possible values reaches its maximal value $\log_2 k$ when the variable is distributed uniformly. This observation will be used to maximize the entropy.

The above optimization constraint is not sufficient to fully specify a risk indicator. To remove degrees of freedom, first note that defining an indicator amounts to defining a partition of the possible values of the metric into 4 components and associating each component with a risk level. Under the assumption that the risk $R(M)$ increases or decreases with M, the partition can be chosen as four consecutive intervals $I_1,\ldots,I_4$. If the risk increases (resp. decreases), a risk level is then assigned to a metric value m as follows:

- green (resp. black) if $m\in I_1$,
- yellow (resp. red) if $m\in I_2$,
- red (resp. yellow) if $m\in I_3$, and
- black (resp. green) if $m\in I_4$.

For example, the risk associated to the proportion iwa4 of files maintained by a single committer naturally increases with this proportion as the FlOSS endeavor becomes more dependent on few committers and thus vulnerable to their possible departure. In the next section, we classify each community metric depending on whether its risk indicator is assumed to be increasing, decreasing or neither.

When it is not reasonable to assume that an indicator is increasing or decreasing, we propose to define it by partitioning the metric values into seven consecutive intervals $I_1,\ldots,I_7$ and assigning the following risk level to a metric value m :

- black if $m\in I_1\cup I_7$,
- red if $m\in I_2\cup I_6$,
- yellow if $m\in I_3\cup I_5$,
- and green if $m\in I_4$.

The rationale behind this constraint is that the central interval I_4 should contain relatively safe metric values and that risk increases with the distance from this interval. As discussed below, we believe that this is a reasonable assumption for metrics such that the maximal entropy indicator has a uniform distribution.

3.2 Maximization of the indicator entropy

To maximize the indicator entropy under the constraints presented above, we use

- a brute-force approach where the optimal indicator partition is found by enumerating all the possible solutions and
- the notion of quantile.

The brute-force approach guarantees an optimal indicator, but it is only applicable when the metric has a limited number of distinct values observed in the data leading to a manageable number of candidate solutions. When there are too many solutions to enumerate in a reasonable time, we choose the quantile approach described as follows.

Let M be a random variable with cumulative distribution function (CDF) $F(m)=P(M\leqslant m)$ and let p be a probability value such that $0<p<1$. If there is one and only one value m of M such that $F(m)=p$, i.e. F is invertible at p, then m is denoted $F^{-1}(p)$ and is called the *p-th quantile*. Hence, provided

M has an invertible CDF at $p=0.25$, $p=0.5$, and $p=0.75$, its values are partitioned by the four intervals $]-\infty, F^{-1}(0.25)]$, $]F^{-1}(0.25), F^{-1}(0.5)]$, $]F^{-1}(0.5), F^{-1}(0.75)]$, and $]F^{-1}(0.75), +\infty[$ with equal probability. Similarly, a partition of the metric values into four equal probability sets is given by the intervals $]-\infty, F^{-1}(0.125)] \cup]F^{-1}(0.875), +\infty[$, $]F^{-1}(0.125), F^{-1}(0.25)] \cup]F^{-1}(0.5), F^{-1}(0.875)]$, $]F^{-1}(0.25), F^{-1}(0.375)] \cup]F^{-1}(0.625), F^{-1}(0.75)]$, and $]F^{-1}(0.375), F^{-1}(0.625)]$, if the aforementioned quantiles are well-defined. Since the entropy is maximal when the indicator has a uniform distribution, the above partitions are optimal.

In practice, we do not know if the CDF is invertible and we also have to estimate quantiles from data. We propose to estimate quantiles with the formula

$$\hat{F^{-1}}(p) = m_{\lfloor r \rfloor} + (r - \lfloor r \rfloor)(m_{\lfloor r \rfloor} - m_{\lfloor r \rfloor}), \quad (3)$$

where $m_1, \ldots, m_{N-1}$ is the sequence of observed metrics ordered by increasing value and $r = p(N-1)$. Note that Equation (3) is defined even if the CDF $F(m)$ is not invertible at p. If $F(m)$ is well-behaved, then $\hat{F^{-1}}(p)$ will converge to the quantile $F^{-1}(p)$ as the sample size increases[2]. The estimated quantiles are then used to define interval bounds for the indicator partition. Although the resulting indicator may not be optimal, this optimization procedure performed reasonably well in our experiments.

It is important to realize that our max entropy indicators define *relative risks*, and not *absolute risks*. For an increasing or decreasing indicator, a low risk does not imply that a metric value is intrinsically good, just good compared to other projects. For instance, a green value for a decreasing indicator should be interpreted as a statement that the corresponding metric is in the top quartile. For an indicator that is neither increasing nor decreasing, a risk should be interpreted as a distance from the median behavior. For instance, a low risk (green) means that the metric is close to its median value, while a high risk (black) means that the metric value is uncommon. Given that defining thresholds on the risk of FlOSS community behavior is new, using such relative indicators seems to be a reasonable approach.

3.3 Application to community metrics using FLOSSMetrics data

In this deliverable, we choose to classify the increasing or decreasing character of indicators defined with the community metrics as indicated in Table 1. To some degree, the classification is arbitrary. For example, one may agree that the risk associated to the slope of the number of new code committers (sra2) generally decreases with increasing value of sra2, but argue that this no longer holds for very high sra2 as having a large number of new committers may destabilize the code base. However, in practice, it has rarely been observed that the number of new committers grows to a point where the code based is jeopardized. Thus, we feel that the choices made in Table 1 are reasonable.

[2]Technically, if F is a homeomorphism around $F^{-1}(p)$, then $\hat{F^{-1}}(p)$ converges to the corresponding quantile with probability one as the sample size goes to infinity.

Indicator	Type
slope of the number of new code committers (sra2)	decreasing
slope of the number of new non-code committers (sra3)	decreasing
slope of the number of active code committers (sra9)	decreasing
slope of the ratio of number of commits by number of committers (iwa1)	decreasing
slope of the ratio of number of code commits by number of code committers (iwa2)	decreasing
slope of the number of new core committers (sra4)	decreasing
slope of the number of core committers leaving (sra5)	increasing
slope of the difference between sra4 and sra5	decreasing
average number of month where each committer committed (sra7)	decreasing
proportion of files maintained by a single committer (iwa4)	increasing
ratio of the number of lines of code by the number of active committers (iwa5)	neither
proportion of code files committed to in the last year (iwa7)	neither

*Table 1: C*lassification of each indicator as a function of its metric that increases, decreases, or neither.

To compute metrics and design indicators, we retrieved data collected by the FLOSSMetrics project. Let us note that the data provided by FLOSSMetrics is still growing and evolving. Many of the open-source projects considered by FLOSSMetrics appear to be very small and are thus not representative of our population of interest. Indeed, we are assessing the risks associated to a full floss collaboration with open-source projects. This precludes very small projects for which, from a business perspective, a fork should be more appropriate. Hence, we choose to filter out the FLOSSMetrics data projects where less than four distinct committers committed over the entire project life, as suggested in [1]. As discussed below, this particular choice of filter appears to increase the quality of our indicators. The effect of filtering on the number of projects with sufficient data in FLOSSMetrics to measure each metric is given in Table 2.

Metric	Interval length	Unfiltered data	Filtered data
sra7	-	1422	735
iwa4	-	1422	735
iwa5	-	676	289
iwa7	-	679	291
sra2	30 days	512	291
sra2	90 days	500	289
sra2	180 days	470	285
sra3	30 days	512	291
sra3	90 days	500	289
sra3	180 days	470	285
sra9	30 days	512	291
sra9	90 days	500	289
sra9	180 days	470	285
iwa1	30 days	1180	735
iwa1	90 days	1152	732
iwa1	180 days	1095	727
iwa2	30 days	512	291
iwa2	90 days	500	289
iwa2	180 days	470	285
sra4	1 year	1119	729
sra5	1 year	1119	729
sra6	1 year	1119	729

Table 2: Number of projects with sufficient data in FLOSSMetrics to measure each metric.

All of the metrics considered have a number of distinct values observed in the filtered FLOSSMetrics dataset that is too large to maximize the entropy by an exhaustive search. Hence, we use the quantiles approach and we obtain with the indicators given in Table 3, Table 4, and Table 5 These tables also incorporate some natural constraints on metrics implied by their definition: sra7 $\geqslant$, $0\leqslant$ iwa4 $\leqslant$, iwa5 $\geqslant 0$, and $\bullet\leqslant$ iwa7 $\leqslant 1$. Note that slope metrics have no such constraints.

Metric	Indicator			
	Black	Red	Yellow	Green
sra7	$[1,5.53555]$	$]5.53555,8.5789]$	$]8.5789,12.25]$	$]12.25,+\infty]$
iwa4	$]٠.٨٧٢٤٥,+\infty]$	$]٠.٧١٢٤,٠.٨٧٢٤٥]$	$]0.5522,٠.٧١٢٤]$	$[٠,٠.٥٥٢٢]$
iwa5	$[٠,٢١٣٠.٧]$ $\cup]٧٢٢٩٥,+\infty]$	$]2130.7,5293.5]$ $\cup]٤٦٠٢٩.١,٧٢٢٩٥]$	$]5293.5,٩٧٩١.٧]$ $\cup]29543.5,46029.1]$	$]9791.7,29543.5]$
iwa7	$[0,0.012875]$ $\cup]٠.٣٦٤٤٢٥,١]$	$]0.012875,٠.٠٤٤٣]$ $\cup]٠.٢٥٥٥,0.364425]$	$]0.0443,0.076625]$ $\cup]0.188075,٠.٢٥٥٥]$	$]٠.٠٧٦٦٢٥,0.188075]$

Table 3: Indicators for the static metrics sra7, iwa4, iwa5, and iwa7.

Metric	Interval length	Indicator			
		Black	Red	Yellow	Green
sra2	30 days	$]-\infty,-0.01]$	$]-0.01,-٠.٠٠٤٤]$	$]-٠.٠٠٤٤,-0.0013]$	$]-٠.٠٠١٣,+\infty[$
sra2	90 days	$]-\infty,-٠.٠٨٣٣]$	$]-٠.٠٨٣٣,-٠.٠٣٨٧]$	$]-0.0387,-0.0129]$	$]-٠.٠١٢٩,+\infty]$
sra2	180 days	$]-\infty,-0.3167]$	$]-٠.٣١٦٧,-0.1469]$	$]-٠.١٤٦٩,-٠.٠٥٢٧]$	$]-٠.٠٥٢٧,+\infty[$
sra3	30 days	$]-\infty,-0.0224]$	$]-٠.٠٢٢٤,-٠.٠٠٧٨]$	$]-0.0078,-0.0034]$	$]-٠.٠٠٣٤,+\infty]$
sra3	90 days	$]-\infty,-0.1993]$	$]-٠.١٩٩٣,-٠.٠٦٨٤]$	$]-0.0684,-0.0296]$	$]-٠.٠٢٩٦,+\infty[$
sra3	180 days	$]-\infty,-0.7]$	$]-0.7,-٠.٢٦٤٧]$	$]-0.2647,-0.1135]$	$]-0.1135,+\infty[$
sra9	30 days	$]-\infty,-٠.٠٢٣]$	$]-٠.٠٢٣,-٠.٠٠٥]$	$]-0.005,٠.٠١٠٨٥]$	$]٠.٠١٠٨٥,+\infty[$
sra9	90 days	$]-\infty,-٠.٠٩٠٩]$	$]-٠.٠٩٠٩,-٠.٠٢٤٣]$	$]-٠.٠٢٤٣,٠.٠٤٢]$	$]٠.٠٤٢,+\infty[$
sra9	180 days	$]-\infty,-٠.٢٣٦٤]$	$]-٠.٢٣٦٤,-٠.٠٥٩٣]$	$]-0.0593,٠.٠٩٠١]$	$]٠.٠٩٠١,+\infty[$
iwa1	30 days	$]-\infty,-0.5418]$	$]-٠.٥٤١٨,-٠.٠٤٥٠]$	$]-0.0450,٠.٢٤٠١]$	$]0.2401,+\infty[$
iwa1	90 days	$]-\infty,-4.0177]$	$]-٤.٠١٧٧,-0.5290]$	$]-0.5290,١.٣٤١١]$	$]١.٣٤١١,+\infty[$
iwa1	180 days	$]-\infty,-١٣.٤٠٤٧]$	$]-13.4047,-1.5654]$	$]-1.5654,4.1337]$	$]4.1337,+\infty[$
iwa2	30 days	$]-\infty,-٠.٧٩٤٤]$	$]-٠.٧٩٤٤,-0.1505]$	$]-٠.١٥٠٥,٠.٠٥١٩]$	$]0.0519,+\infty[$
iwa2	90 days	$]-\infty,-5.5961]$	$]-٥.٥٩٦١,-١.٠٠٥٦]$	$]-1.0056,0.3397]$	$]٠.٣٣٩٧,+\infty[$
iwa2	180 days	$]-\infty,-19.4]$	$]-١٩.٤,-3.6696]$	$]-3.6696,٠.٨٧٣٥]$	$]٠.٨٧٣٥,+\infty[$

Table 4: Indicators for the slope metrics sra2, sra3, sra9, iwa1, and iwa2.

Metric	Indicator			
	Black	Red	Yellow	Green
sra4	$]-\infty,-٠.٥]$	$]-0.5,-0.2]$	$]-0.2,-0.0357]$	$]-٠.٠٣٥٧,+\infty[$
sra5	$]٠.٤,+\infty[$	$]٠.١,٠.٤]$	$]0,٠.١]$	$]-\infty,٠]$
sra6	$]-\infty,-٠.٧٦٣٦]$	$]-0.7636,-٠.٣]$	$]-٠.٣,-0.1429]$	$]-0.1429,+\infty[$

Table 5: Indicators for the slope metrics sra4, sra5, and sra6.

Provided the metric CDF is invertible at our points of interest and there is sufficient data, each risk level should have approximately the same number of projects by construction. To assess the validity of each indicator, we thus present the number of projects falling into each risk level in Table 6, Table 7, and Table 8. Overall the projects seem to be evenly distributed across the risk levels for all the metrics except the slopes of the number of new core committers (sra4), of the number of core committers leaving (sra5), and of their difference (sra6). We believe that this phenomenon is explained by the fact that many projects have short duration and the slope of the metrics sra4, sra5 and sra6 is computed based on very few points. As a result, some slope values become very common, leading to non-invertible CDFs. In particular, there are 149 projects with a zero value for sra4 and 34 projects with a value of -0.3 for sra6. Since the risk distributions for sra4, sra5, and sra6 are not uniform, there is no guarantee that they maximize the entropy. Unfortunately, it is not realistic to use the brute-force approach to find optimal indicators as there are too many possible solutions to score. There may exist more clever optimization techniques able to solve this issue, but we believe that data with longer projects would provide a better solution.

	Indicator				
Metric	Black	Red	Yellow	Green	Total
sra7	184	184	185	182	735
iwa4	184	183	184	184	735
iwa5	73	72	72	72	289
iwa7	74	72	72	73	291

Table 6: Counts of the observed indicator values for the static metrics sra7, iwa4, iwa5, and iwa7.

		Indicator				
Metric	Interval length	Black	Red	Yellow	Green	Total
sra2	30 days	73	74	74	70	291
sra2	90 days	75	70	72	72	289
sra2	180 days	72	71	71	71	285
sra3	30 days	73	74	73	71	291
sra3	90 days	73	72	72	72	289
sra3	180 days	73	70	71	71	285
sra9	30 days	74	72	72	73	291
sra9	90 days	73	72	72	72	289
sra9	180 days	72	71	71	71	285
iwa1	30 days	184	184	183	184	735
iwa1	90 days	183	183	183	183	732
iwa1	180 days	182	182	181	182	727
iwa2	30 days	73	73	72	73	291
iwa2	90 days	73	72	72	72	289
iwa2	180 days	72	71	71	71	285

Table 7: Counts of the observed indicator values for the slope metrics sra2, sra3, sra9, iwa1, and iwa2.

	Indicator				
Metric	Black	Red	Yellow	Green	Total
sra4	199	171	177	182	729
sra5	174	172	114	269	729
sra6	183	214	157	175	729

Table 8: Counts of the observed indicator values for the slope metrics sra4, sra5, and sra6.

To illustrate the benefit of filtering, let us consider the construction of the indicator for the slope of the ratio of the number of commits by the number of committers (iwa4) using the original data. Using Equation (3), we obtain $\hat{F}^{-1}(0.25) = 0.693$, $\hat{F}^{-1}(0.5) = 0.933750$, and $\hat{F}^{-1}(0.75) = 1$. Moreover, there are 356 projects in the interval $]-\infty, 0.693]$, 355 in $]0.693, 0.93375]$, 711 in $]0.93375, 1]$, and none in $]1, +\infty]$. In fact, there are 496 projects with iwa4 at 1, representing 34.88% of the projects. Hence, it is not possible to partition the projects into four sets with approximately equal size. After filtering, note that only two projects have the value 1. The resulting risk distribution is almost uniform(see Table 6) and thus maximizes the entropy.

4. Beyond slope metrics: outline of a predictive approach

4.1 Shortcomings of the previous approach

The indicators presented above measure relative risks and not absolute risks. This is acceptable for metrics that are easy to interpret such as the static metrics (sra7, iwa4, iwa5, and iwa7). For instance, it is intuitively plausible to say that the risk level associated to the average longevity of committers (sra7) is green if sra7 is more than a year. On the other hand, the situation is less satisfying for slope metrics and we find it difficult to justify intuitively our choice of indicators. For instance, one may ask why the indicator associated to sra2 and with a 90 days interval length is red if $-0.0833 <$ sra2 $\leqslant -0.0387$, but yellow if $-0.0387 <$ sra2 $\leqslant -0.0129$. It is not straightforward to convince oneself that these intervals do correspond to different risk levels.

To help identify the shortcomings of our slope indicators, consider the sequences of the number of new code committers (low-level metric sra2) computed over intervals of, respectively, 30, 90, and 180 days and their linear approximations given in, respectively, Illustration 1, Illustration 2, and Illustration 4 for the open-source project evolution. The following issues become readily apparent:

- although its sign is intuitive, the absolute value of a slope is hard to interpret,
- using a slope metric implies the construction of a linear model which may not be adequate to model the sequence of metric values and whose slope may be a crude trend measure, and
- there is no guidance to pick the interval length over which the low-level metric is computed.

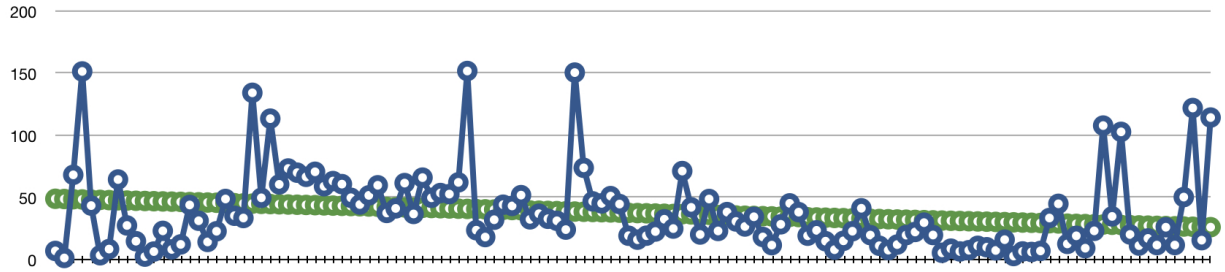

Illustration 1: Evolution of sra2 and approximating linear model with 30 days intervals.

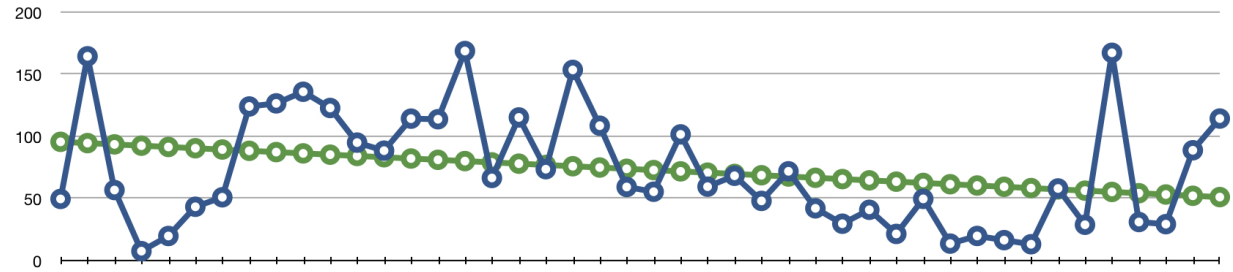

Illustration 2: Evolution of sra2 and approximating linear model with 90 days intervals.

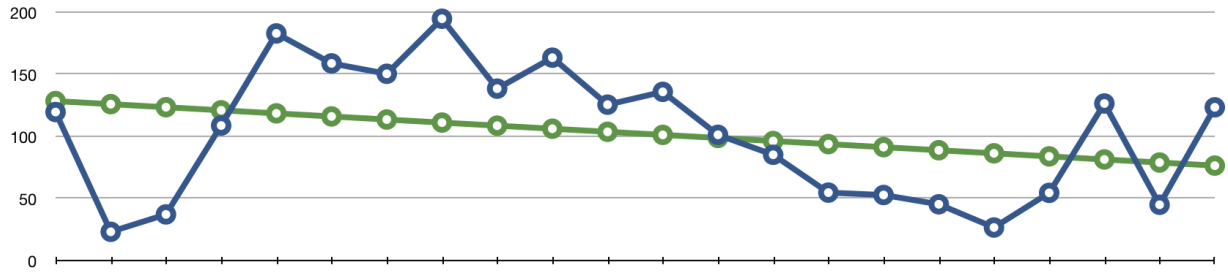

Illustration 3: Evolution of sra2 and approximating linear model with 180 days intervals.

4.2 A PREDICTIVE FORMULATION OF RISK ASSESSMENT

Let us analyse slope metrics to gain some insight and devise better indicators. To compute a slope metric, recall that we divide the lifespan of the project into a sequence of consecutive time intervals $t_1,..,t_n$ of length T, compute a sequence of low-level metric values, model this sequence by a least square regression line $y(t)=w_0+w_1 t/T$, and finally use the slope w_1 as a metric. Observe that this slope may be written as the difference between the model values at two consecutive time intervals

$$y(t+T)-y(t)=w_1 \quad (4)$$

In effect, we are using a linear model to predict a difference between two consecutive metric values.

This suggest to generalize our definition of metrics by formulating our risk assessment problem as a prediction problem where we

- decide on the quantity to predict: in addition to absolute metric differences, we propose to predict , absolute metric values and relative differences between consecutive metric values some time in the future, and
- choose a model to predict the above quantity based on prior knowledge or available data, e.g. build a linear model such as a sequence of low-level metric values from a project.

The proposed predictive methodology has several benefits.

- The indicators only depend on the predicted quantity and no longer on the choice of prediction model. If necessary, they can still be obtained by the methodology described in Section 3.
- The choice of model to predict the above quantity may be generalized: advanced models tailored to the choice of prediction may be tested, some may also give a confidence measure in their prediction.
- The predicted quantity has a simple interpretation, unlike slope metrics.

In particular, there is now a clear separation between the interval length used in the linear model of a sequence and the length of time between the current time and the time for which the prediction is made.

4.3 APPLICATION OF THE PREDICTIVE APPROACH USING FLOSSMETRICS DATA

Using FLOSSMetrics data, let us design community risk indicators for predicted absolute metric values, absolute metric differences, and relative metric differences. Before starting a full floss collaboration with a project, a company has to pick a time horizon over which risks should be assessed, based on its business objective. This time horizon determines the choice of time interval for the prediction. For example, we may want to predict the evolution of new code committers after three or six months. If the interval is too short, say one month, the prediction becomes useless from a business perspective. If the interval is too long, it may become impossible to estimate the evolution accurately. In this deliverable, we consider an interval length for prediction of one year for sra4, sra5 and sra6. For sra2, sra3, sra9, iwa1, and iwa2, we consider 90 and 180 days.

Let us define new indicators by applying the max entropy methodology presented in Section 3.1. First, let us describe the data used. As before, given an interval length T and a project, we divide the lifespan of the project into a sequence of consecutive time intervals $t_1, .., t_n$ of length T. Given such a sequence and a low-level metric m, we build

- the sequence $m(t_1), ..., m(t_n)$ of absolute metric values computed over each interval,
- the sequence $m(t_2)-m(t_1), ..., m(t_n)-m(t_{n-1})$ of absolute metric differences between consecutive intervals, and
- the sequence $(m(t_2)-m(t_1))/|m(t_1)|, ..., (m(t_n)-m(t_{n-1}))/|m(t_{n-1})|$ of relative metric differences between consecutive intervals, where $0/0$ is treated as 0 and $+\infty$ is a valid measure.

For each low-level metric and time interval length, we thus obtain a dataset of

- absolute metric values,
- absolute metric differences, and
- relative metric differences

by pooling each of the associated sequence for all of the projects in the FLOSSMetrics database that have more than four committers. Let us design indicators using these datasets.

4.3.1 Absolute metric values

The ratio iwa1 (resp. iwa2) of the number of commits (resp. code commits) by the number of committers (resp. code committers) have a number of distinct values observed in the data that is too large to design max-entropy indicators by brute-force optimization. Hence, we design their indicators by estimating quantiles. On the other hand, the number of new code committers (sra2), the number of new non-code committers (sra3), the number of new core committers (sra4), the number of core committers that leave (sra5), the difference between sra4 and sra5 (sra6), and the number of active code committers (sra9) each have a small number of possible values observed allowing an exact brute-force entropy maximization. Table 9 presents the resulting indicators. Table 10 presents the number of observed indicator values in the dataset. When the number of possible metric values is small, it is possible to visualize indicators with histograms of the observed metric values colored by risk level. Illustration 4 to Illustration 12 present such histograms for sra2, sra3, sra4, sra5, sra6, and sra9.

Metric	Prediction interval length	Indicator			
		Black	Red	Yellow	Green
sra2	90 days	0	1	2	$[3,+\infty[$
sra2	180 days	0	١	$[2,٣]$	$[٤,+\infty[$
sra3	90 days	٠	1	$[٢,5]$	$[٦,+\infty[$
sra3	180 days	٠	١	$[2,٧]$	$[8,+\infty[$
sra9	90 days	0	١	$[2,3]$	$[4,+\infty[$
sra9	180 days	$[0,١]$	2	$[3,٤]$	$[5,+\infty[$
iwa1	90 days	$[0,6.5[$	$[6.5,32.6[$	$[32.6,١٠٧.٥[$	$[107.5,+\infty[$
iwa1	180 days	$[٠,١٢.١[$	$[12.1,60.8]$	$]٦٠.٨,187.1[$	$[187.1,+\infty[$
iwa2	90 days	$[0.0,٣.٣]$	$]3.3,21]$	$]21,76]$	$]76,+\infty[$
iwa2	180 days	$[٠.٠,8.75[$	$[8.75,42[$	$[42,133.7[$	$[133.7,+\infty[$
sra4	1 year	٠	1	2	$[٣,+\infty[$
sra5	1 year	$[4,+\infty[$	$[٢,٣]$	1	٠
sra6	1 year	$]-\infty,-1]$	0	1	$[٢,+\infty[$

Table 9: Max-entropy indicators for the absolute metric values.

Metric	Prediction interval length	Indicator				
		Black	Red	Yellow	Green	Total
sra2	90 days	3792	1182	455	529	5958
sra2	180 days	1473	670	531	375	3049
sra3	90 days	3076	858	1081	943	5958
sra3	180 days	1187	499	697	666	3049
sra9	90 days	900	1682	2026	1350	5958
sra9	180 days	980	635	717	717	3049
iwa1	90 days	3838	3862	3849	3852	15401
iwa1	180 days	1973	1973	1972	1973	7891
iwa2	90 days	1492	1497	1484	1485	5958
iwa2	180 days	762	760	764	763	3049
sra4	1 year	1850	1331	577	737	4495
sra5	1 year	2487	1072	512	424	4495
sra6	1 year	1112	1641	966	776	4495

Table 10: Counts of the observed indicator values for the absolute metric values.

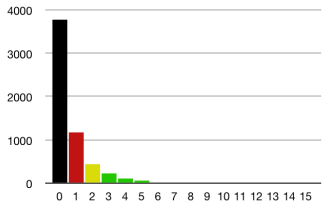

Illustration 4: sra2, 90 days

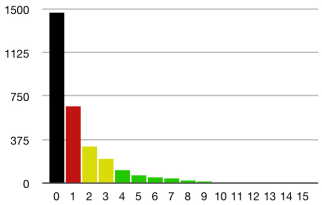

Illustration 5: sra2, 180 days

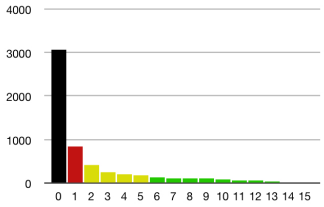

Illustration 6: sra3, 90 days

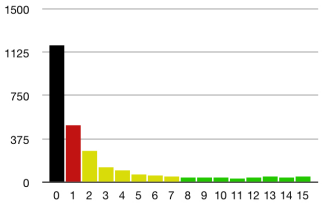

Illustration 7: sra3, 180 days

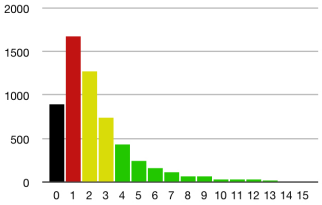

Illustration 8: sra9, 90 days

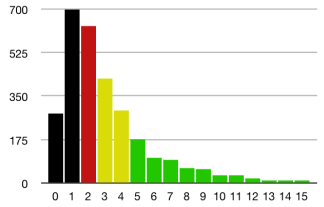

Illustration 9: sra9, 180 days

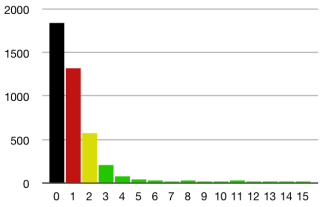

Illustration 10: sra4

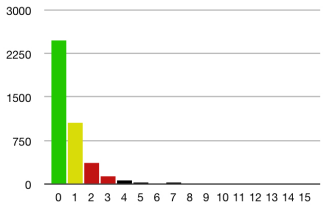

Illustration 11: sra5

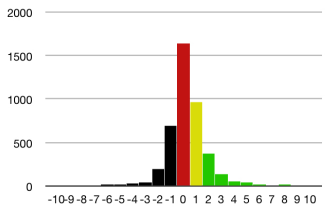

Illustration 12: sra6

4.3.2 Absolute metric differences

Again, the indicators for iwa1 and iwa2 are obtained using quantiles, while the indicators for sra2, sra3, sra4, sra5, sra6 and sra9 are obtained by brute-force maximization. Table 11 presents the resulting max entropy indicators. Table 12 presents the counts of the risk levels observed in the data. Illustration 14 to Illustration 21 display histograms of the observed metric values colored by risk level for sra2, sra3, sra4, sra5, sra6, and sra9.

Metric	Prediction interval length	Indicator			
		Black	Red	Yellow	Green
sra2	90 days	$]-\infty,-2]$	-1	0	$[1,+\infty[$
sra2	180 days	$]-\infty,-2]$	-1	0	$[١,+\infty[$
sra3	90 days	$]-\infty,-2]$	-1	٠	$[١,+\infty[$
sra3	180 days	$]-\infty,-3]$	$[-2,-1]$	0	$[١,+\infty[$
sra9	90 days	$]-\infty,-2]$	-1	٠	$[1,+\infty[$
sra9	180 days	$]-\infty,-٢]$	-1	0	$[١,+\infty[$
iwa1	90 days	$]-\infty,-١٩.٣۶[$	$[-19.36,٠[$	$[0,14.5[$	$[14.5,+\infty[$
iwa1	180 days	$]-\infty,-٣٩]$	$]-39,-١]$	$]-1,23.68]$	$]٢٣.۶٨,+\infty[$
iwa2	90 days	$]-\infty,-20.28]$	$]-٢٠.٢٨,0[$	$[0,١٣.۵[$	$[13.5,+\infty[$
iwa2	180 days	$]-\infty,-٣٧.٢٢]$	$]-37.22,-2]$	$]-٢,١٨.۵]$	$]18.5,+\infty[$
sra4	1 year	$]-\infty,-2]$	-١	0	$[١,+\infty[$
sra5	1 year	$[٢,+\infty[$	١	٠	$]-\infty,-١]$
sra6	1 year	$]-\infty,-٢]$	-١	0	$[1,+\infty[$

Table 11: Max-entropy indicators for the absolute differences between consecutive metric values.

Metric	Prediction interval length	Indicator				
		Black	Red	Yellow	Green	Total
sra2	90 days	521	875	3121	1150	5667
sra2	180 days	468	486	1125	679	2758
sra3	90 days	992	689	2591	1395	5667
sra3	180 days	510	601	960	687	2758
sra9	90 days	709	982	2292	1684	5667
sra9	180 days	453	481	953	871	2758
iwa1	90 days	3667	3644	3684	3671	14666
iwa1	180 days	1792	1787	1788	1789	7156
iwa2	90 days	1417	1320	1509	1421	5667
iwa2	180 days	690	692	689	687	2758
sra4	1 year	610	828	1432	890	3760
sra5	1 year	888	1491	829	552	3760
sra6	1 year	960	653	1059	1088	3760

Table 12: Counts of the observed indicator values for the absolute differences between consecutive metric values.

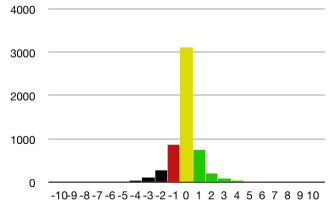

Illustration 13: sra2, 90 days.

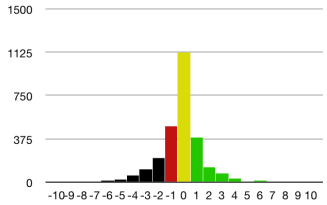

Illustration 14: sra2, 180 days.

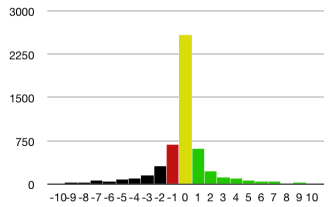

Illustration 15: sra3, 90 days.

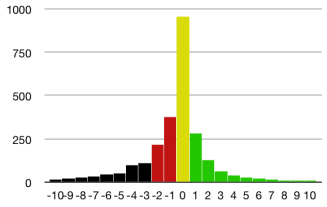

Illustration 16: sra3, 180 days.

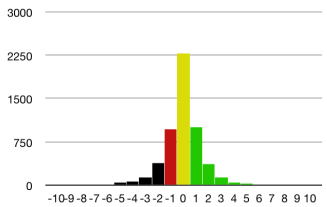

Illustration 17: sra9, 90 days.

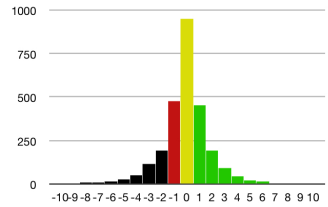

Illustration 18: sra9, 180 days.

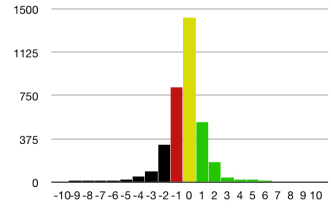

Illustration 19: sra4.

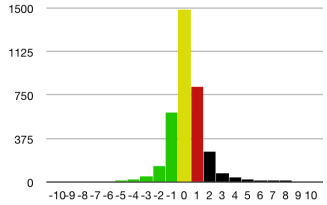

Illustration 20: sra5.

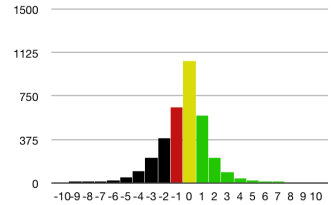

Illustration 21: sra6.

4.3.3 Relative metric differences

Relative differences have a meaningful scale and have a range of values between -1 and $+\infty$. Consequently, we propose to use the following definition for all of the increasing indicators.

- black if $m \in [-1, -0.5[$,
- red if $m \in [-0.5, 0[$,
- yellow if $m \in [0, 0.5[$,
- green if $m \in [0.5, +\infty]$.

For the decreasing indicator associated to the number of core committers that quit (sra5), we use

- green if $m \in [-\infty, -0.5[$,
- yellow if $m \in [-0.5, 0[$,
- red if $m \in [0, 0.5[$,
- black if $m \in [0.5, +\infty]$.

Table 13 contains the counts of the observed risk levels in the filtered FLOSSMetrics dataset.

Metric	Prediction interval length	Indicator				
		Black	Red	Yellow	Green	Total
sra2	90 days	1142	254	3159	1112	5667
sra2	180 days	736	218	168	636	2758
sra3	90 days	1223	458	2775	1211	5667
sra3	180 days	671	440	1124	523	2758
sra9	90 days	611	1080	2646	1330	5667
sra9	180 days	284	650	1195	629	2758
iwa1	90 days	3289	4022	3385	3970	14666
iwa1	180 days	1623	2228	1494	1811	7156
iwa2	90 days	1685	1052	1138	1792	5667
iwa2	180 days	830	661	477	790	2758
sra4	1 year	1063	375	1512	810	3760
sra5	1 year	1302	1570	222	666	3760
sra6	1 year	1264	87	1092	1317	3760

Table 13: Counts of the observed indicator values for the relative differences between consecutive metric values.

5. Illustration on selected projects

In this section, we compute the metrics and indicators defined in this deliverable for several open-source projects: findbugs, evolution, evince, nautilus, and httpd 1.3. All of these projects are part of the snapshot of the FLOSSMetrics data used to compute our metrics and indicators, except httpd 1.3 that was added at a later date. First, Section 5.1 presents the values of the static metrics and indicators. Then, Section 5.2 presents the values of the slope metrics and indicators. Finally, Section 5.3 presents an application of the predictive approach to risk assessment.

5.1 Static metrics and indicators

As described in Section 2.2, the static metrics measure the average longevity of committers (sra7), the proportion of files maintained by a single committer (iwa4), the ratio of the number of lines of source code by the number of committers active in the last year (iwa5), and the proportion of code files committed to in the last year.(iwa7). When there is sufficient data available to compute them, the values of these static metrics and the corresponding risk levels obtained with Table 3 are given in Table 14.

Metric	findbugs	evolution	evince	nautilus	httpd1.3
sra7	11.7857 Y	8.6044 Y	5.0272 B	7.4484 R	15.7206 G
iwa4	0.8779 B	0.3666 G	0.4216 G	0.3168 G	0.0923 G
iwa5	- -	374772.4000 B	48282.8000 R	246991.9000 B	- -
iwa7	- -	0.1450 G	0.1904 Y	0.0491 Y	0.0006 B

Table 14: Static metric and risk indicator values.

Overall, the above results are easy to interpret and satisfying. For example, it is not counterintuitive to state that httpd has good values for sra7 and iwa4 and thus a low risk. When interpreting the results for iwa5 and iwa7, keep in mind that the indicator color indicates a distance from the median metric value. For example, the black color associated to the iwa7 value for httpd means that its iwa7 value is uncommon, not intrinsically risky.

5.2 Slope metrics and indicators

As described in Section 2.1, slope metrics measure the evolution of the number of new code committters (sra2), the number of new non-code committers (sra3), the number of active code committers (sra9), the ratio of the number of commits by the number of committers (iwa1), the ratio of the number of code commits by the number of code committers (iwa2), the number of new core committers (sra4), the number of core committers leaving, and the difference sra6 between sra4 and sra5.

Recall that to compute a slope metric, the lifespan of a project is first divided into a sequence of consecutive time intervals $I_1, \ldots, I_n$ of length T covering all the commit activity. Then, a linear model of the metric sequence $m(I_1), \ldots, m(I_n)$ is estimated from data: $y(t) = w_0 + w_1 t/T$. The slope w_1 is then defined as our slope metric. For each of the selected project and each of the interval length in 30 days, 90 days, 180 days and 1 year, Table 15 presents the number of time intervals necessary to cover the project lifespan. When there is sufficient data available to compute them, Table 16 to Table 20 give the intercept and slope of the linear models estimated by least square regression using the FLOSSMetrics data for each of the selected projects. These tables also contain the indicator values computed from the slopes using the intervals given in Table 4 and Table 5.

As discussed in Section 4.1, we are not confident that the indicator values assess risks appropriately. Consider in particular the indicators and metrics associated to sra2, sra3, and sra9: the slopes of the linear models estimated from data are suspiciously close to zero. While the indicator values computed with these slopes represent a valid ranking for each metric, we do not believe that this ranking reflects true risk levels

Project	Interval length	Number of intervals
findbugs	30 days	61
findbugs	90 days	21
findbugs	180 days	11
findbugs	1 year	6
evolution	30 days	130
evolution	90 days	44
evolution	180 days	22
evolution	1 year	11
evince	30 days	115
evince	90 days	39
evince	180 days	20
evince	1 year	10
nautilus	30 days	129
nautilus	90 days	43
nautilus	180 days	22
nautilus	1 year	11
httpd 1.3	30 days	166
httpd 1.3	90 days	56
httpd 1.3	180 days	28
httpd 1.3	1 year	14

Table 15: Number of intervals used to cover the lifespan of the selected projects.

Metric	Linear model interval length	Intercept	Slope	
iwa1	30 days	310.0719	0.0000	G
iwa1	90 days	820.7061	0.0000	G
iwa1	180 days	1154.0980	0.0000	G
sra4	1 year	0.0952	-0.2286	R
sra5	1 year	0.7143	0.0000	Y
sra6	1 year	-0.6191	0.0000	R

Table 16: Parameters of the linear models and slope indicator values for findbugs.

Metric	Linear model interval length	Intercept	Slope	
sra2	30 days	0.8625	-0.0106	B
sra2	90 days	2.4010	-0.1008	B
sra2	180 days	4.9805	-0.3958	B
sra3	30 days	2.5806	-0.0071	Y
sra3	90 days	7.2788	-0.0790	R
sra3	180 days	14.8182	-0.2987	R
sra9	30 days	13.4552	0.0129	G
sra9	90 days	19.6890	0.0257	Y
sra9	180 days	26.0596	-0.0203	Y
iwa1	30 days	5.3706	-0.0382	Y
iwa1	90 days	8.8816	-0.2333	Y
iwa1	180 days	13.2741	-0.6177	Y
iwa2	30 days	26.6236	-0.1772	R
iwa2	90 days	51.6623	-1.0345	R
iwa2	180 days	77.0011	-2.4736	Y
sra4	1 year	10.2273	0.5000	G
sra5	1 year	12.6365	1.2364	B
sra6	1 year	-2.4093	-0.7364	R

Table 17: Parameters of the linear models and slope indicator values for evolution.

Metric	Linear model interval length	Intercept	Slope	
sra2	30 days	0.7117	0.0044	G
sra2	90 days	2.0164	0.0346	G
sra2	180 days	3.6143	0.1015	G
sra3	30 days	3.3070	0.0301	G
sra3	90 days	9.3606	0.2457	G
sra3	180 days	16.9001	0.8158	G
sra9	30 days	3.5727	0.0311	G
sra9	90 days	5.7996	0.1379	G
sra9	180 days	8.2286	0.3609	G
iwa1	30 days	2.8221	0.0058	Y
iwa1	90 days	4.2682	-0.0313	Y
iwa1	180 days	5.3603	-0.2450	Y
iwa2	30 days	22.0781	0.1234	G
iwa2	90 days	37.9864	0.3200	Y
iwa2	180 days	52.5288	-0.1564	Y
sra4	1 year	14.0362	1.5636	G
sra5	1 year	11.5816	1.4848	B
sra6	1 year	2.4546	0.0788	G

Table 18: Parameters of the linear models and slope indicator values for evince.

Metric	Linear model interval length	Intercept	Slope	
sra2	30 days	0.7411	-0.0078	R
sra2	90 days	2.2488	-0.0701	R
sra2	180 days	4.0471	-0.3072	R
sra3	30 days	2.9598	0.0024	G
sra3	90 days	8.8680	0.0214	G
sra3	180 days	16.4073	-0.0045	G
sra9	30 days	5.1040	-0.0226	R
sra9	90 days	10.2007	-0.0735	R
sra9	180 days	14.7510	-0.2445	B
iwa1	30 days	2.0941	-0.0303	Y
iwa1	90 days	2.8426	-0.2016	Y
iwa1	180 days	2.7895	-0.7302	Y
iwa2	30 days	9.2596	-0.2664	R
iwa2	90 days	12.0182	-1.6089	R
iwa2	180 days	8.5524	-5.5044	R
sra4	1 year	12.4546	0.1091	G
sra5	1 year	18.2728	1.5091	B
sra6	1 year	-5.8182	-1.4000	B

Table 19: Parameters of the linear models and slope indicator values for nautilus.

Metric	Linear model interval length	Intercept	Slope	
sra2	30 days	-0.0917	-0.0052	R
sra2	90 days	-0.2733	-0.0463	R
sra2	180 days	-0.4827	-0.1839	R
sra3	30 days	-0.0457	-0.0053	Y
sra3	90 days	-0.1428	-0.0474	Y
sra3	180 days	-0.2315	-0.1891	Y
sra9	30 days	-0.7703	-0.0660	B
sra9	90 days	0.1957	-0.2760	B
sra9	180 days	1.5569	-0.6360	R
iwa1	30 days	-2.0973	-0.0876	R
iwa1	90 days	-6.2849	-0.6211	R
iwa1	180 days	-11.6181	-2.1927	R
iwa2	30 days	6.4280	-0.0894	Y
iwa2	90 days	0.0220	-0.7634	Y
iwa2	180 days	-1.8747	-2.7150	Y
sra4	1 year	-0.1140	-0.3582	R
sra5	1 year	2.6572	0.0132	Y
sra6	1 year	-2.7712	-0.3714	R

Table 20: Parameters of the linear models and slope indicator values for httpd 1.3.

5.3 Predictive risk assessment

Let us use the linear models estimated in the previous section to predict absolute metric values, absolute metric differences, and relative metric differences. Although used here to illustrate the applicability of the methodology developed in this deliverable, we do not necessarily recommend our choice of linear models for prediction as such model may not represent the data adequately.

Let us describe how we use linear models for prediction. In the expression $y(t)=w_0+w_1 t/T$ and in Table 16 to Table 20, the intercept w_0 is chosen so that the time value $t=0$ coincides with the end of the last interval I_n covering the project lifespan. This particular choice simplifies the expression of the predictions. After a time T_p, a linear model may predict that

- the absolute metric value is $y(T_p)=w_0+w_1 T_p/T$,
- the absolute metric difference is $y(T_p)-y(0)=w_1 T_p/T$, and
- the relative metric difference is $y(T_p)-y(0)/|y(0)|=(w_1 T_p)/(|w_0| T)$.

However, some metrics have the following natural constraints that should be incorporated. First, the absolute metric values for sra2, sra3, sra4, sra5, sra9, iwa1, and iwa2 are non-negative. Second, the absolute metric values and absolute metric differences for sra2, sra3, sra4, sra5, sra6, and sra9 are integer-valued and the prediction should be rounded. Finally, the relative metric differences are never less than -1. For sra2, sra3, sra4, sra5 and sra9, we thus predict instead that

- the absolute metric value is $max\{0,\lfloor w_0+w_1 T_p/T+1/2\rfloor\}$,
- the absolute metric difference is $\lfloor w_1 T_p/T+1/2\rfloor$, and

- the relative metric difference is $max\{-1,(w_1 T_p)/(|w_0|T)\}$.

For iwa1 and iwa2, we predict that

- the absolute metric value is $max\{0,(w_0+w_1 T_p/T)\}$,
- the absolute metric difference is $w_1 T_p/T$, and
- the relative metric difference is $max\{-\backslash,(w_1 T_p)/(|w.|T)\}$.

Finally, for sra6, we predict that

- the absolute metric value is $w.+w_1 T_p/T$,
- the absolute metric difference is $w_\backslash T_p/T$, and
- the relative metric difference is $max\{-1,(w_\backslash T_p)/(|w_0|T)\}$.

Table 21 to Table 25 present the prediction results computed for the selected project as described above. The indicator values are obtained using the intervals given in Section 4.3. One can see that, quite often, very small absolute and relative metric differences are predicted .This is due to the fact that slopes of most linear models are very close to zero. On the other hand, as shown by Illustrations 1 to 3, non zero metrics differences occur often in practice, hinting at the inadequacy of linear models. One can also note that predictions and risk levels sometimes vary with the interval length of the linear model, in particular for sra2. In order to select the best linear model, we should estimate the prediction accuracy of each model on another dataset.

Metric	Linear model interval length	Prediction interval length	Predicted metric value	Predicted absolute metric difference	Predicted relative metric difference
iwa1	30 days	90 days	321.2285 G	11.1566 Y	0.0360 Y
iwa1	90 days	90 days	850.2419 G	29.5400 G	0.0360 Y
iwa1	180 days	90 days	1186.8408 G	32.7428 G	0.0284 Y
iwa1	30 days	180 days	332.3852 G	22.3133 Y	0.0720 Y
iwa1	90 days	180 days	879.7778 G	59.0717 G	0.0720 Y
iwa1	180 days	180 days	1219.5836 G	65.4856 G	0.0567 Y
sra4	1 year	1 year	0 B	0 Y	-1.0000 B
sra5	1 year	1 year	1 Y	0 Y	0.1200 R
sra6	1 year	1 year	-1 B	0 Y	-0.5077 B

Table 21: Predicted absolute metric values, absolute metrics differences, and relative metric differences for findbugs and the associated indicators.

Metric	Linear model interval length	Prediction interval length	Predicted metric value	Predicted absolute metric difference	Predicted relative metric difference
sra2	30 days	90 days	1 R	0 Y	-0.0369 R
sra2	90 days	90 days	2 Y	0 Y	-0.0420 R
sra2	180 days	90 days	5 G	0 Y	-0.0397 R
sra2	30 days	180 days	1 R	0 Y	-0.0737 R
sra2	90 days	180 days	2 Y	0 Y	-0.0840 R
sra2	180 days	180 days	5 G	0 Y	-0.0795 R
sra3	30 days	90 days	3 Y	0 Y	-0.0083 R
sra3	90 days	90 days	7 G	0 Y	-0.0109 R
sra3	180 days	90 days	15 G	0 Y	-0.0101 R
sra3	30 days	180 days	3 Y	0 Y	-0.0165 R
sra3	90 days	180 days	7 Y	0 Y	-0.0217 R
sra3	180 days	180 days	15 G	0 Y	-0.0202 R
sra9	30 days	90 days	13 G	0 Y	0.0029 Y
sra9	90 days	90 days	20 G	0 Y	0.0013 Y
sra9	180 days	90 days	26 G	0 Y	-0.0004 R
sra9	30 days	180 days	14 G	0 Y	0.0058 Y
sra9	90 days	180 days	20 G	0 Y	0.0026 Y
sra9	180 days	180 days	26 G	0 Y	-0.0008 R
iwa1	30 days	90 days	5.2560 B	-0.1100 R	-0.0213 R
iwa1	90 days	90 days	8.6483 R	-0.2333 R	-0.0263 R
iwa1	180 days	90 days	12.9652 R	-0.3089 R	-0.0233 R
iwa1	30 days	180 days	5.1414 B	-0.2292 Y	-0.0427 R
iwa1	90 days	180 days	8.4150 B	-0.4666 Y	-0.0525 R
iwa1	180 days	180 days	12.6564 R	-0.6177 Y	-0.0465 R
iwa2	30 days	90 days	26.0920 Y	-0.5315 R	-0.0200 R
iwa2	90 days	90 days	50.6278 Y	-1.0345 R	-0.0200 R
iwa2	180 days	90 days	75.7643 Y	-1.2368 R	-0.0161 R
iwa2	30 days	180 days	25.5605 R	-1.0631 Y	-0.0399 R
iwa2	90 days	180 days	49.5934 Y	-2.0690 R	-0.0400 R
iwa2	180 days	180 days	74.5275 Y	-2.4736 R	-0.0321 R
sra4	1 year	1 year	11 G	1 G	0.0489 Y
sra5	1 year	1 year	14 B	1 R	0.0978 R
sra6	1 year	1 year	-3 B	-1 R	-0.3056 R

Table 22: Predicted absolute metric values, absolute metrics differences, and relative metric differences for evolution and the associated indicators.

Metric	Linear model interval length	Prediction interval length	Predicted metric value	Predicted absolute metric difference	Predicted relative metric difference
sra2	30 days	90 days	1 R	0 Y	0.0000 Y
sra2	90 days	90 days	2 Y	0 Y	0.0172 Y
sra2	180 days	90 days	4 G	0 Y	0.0140 Y
sra2	30 days	180 days	1 R	0 Y	0.0371 Y
sra2	90 days	180 days	2 Y	0 Y	0.0343 Y
sra2	180 days	180 days	4 G	0 Y	0.0281 Y
sra3	30 days	90 days	3 Y	0 Y	0.0273 Y
sra3	90 days	90 days	10 G	0 Y	0.0262 Y
sra3	180 days	90 days	17 G	0 Y	0.0241 Y
sra3	30 days	180 days	3 Y	0 Y	0.0546 Y
sra3	90 days	180 days	10 G	0 Y	0.0525 Y
sra3	180 days	180 days	18 G	1 G	0.0483 Y
sra9	30 days	90 days	4 Y	0 Y	0.0261 Y
sra9	90 days	90 days	6 G	0 Y	0.0238 Y
sra9	180 days	90 days	8 G	0 Y	0.0219 Y
sra9	30 days	180 days	4 Y	0 Y	0.0522 Y
sra9	90 days	180 days	6 G	0 Y	0.0476 Y
sra9	180 days	180 days	9 G	0 Y	0.0439 Y
iwa1	30 days	90 days	2.8394 B	0.0174 Y	0.0062 Y
iwa1	90 days	90 days	4.2369 B	-0.0313 R	-0.0073 R
iwa1	180 days	90 days	5.2378 B	-0.1225 R	-0.0229 R
iwa1	30 days	180 days	2.8568 B	0.0348 Y	0.0123 Y
iwa1	90 days	180 days	4.2057 B	-0.0625 Y	-0.0146 R
iwa1	180 days	180 days	5.1153 B	-0.2450 Y	-0.0457 R
iwa2	30 days	90 days	22.4483 Y	0.3703 Y	0.0168 Y
iwa2	90 days	90 days	38.3064 Y	0.3200 Y	0.0084 Y
iwa2	180 days	90 days	52.4507 Y	-0.0782 R	-0.0015 R
iwa2	30 days	180 days	22.8186 R	0.7405 Y	0.0335 Y
iwa2	90 days	180 days	38.6264 R	0.6400 Y	0.0168 Y
iwa2	180 days	180 days	52.3725 Y	-0.16 Y	-0.0030 R
sra4	1 year	1 year	16 G	2 G	0.1114 Y
sra5	1 year	1 year	13 B	1 R	0.1282 R
sra6	1 year	1 year	3 G	0 Y	0.0321 Y

Table 23: Predicted absolute metric values, absolute metrics differences, and relative metric differences for the evince project and the associated indicators.

Metric	Linear model interval length	Prediction interval length	Predicted metric value	Predicted absolute metric difference	Predicted relative metric difference
sra2	30 days	90 days	1 R	0 Y	-0.0316 R
sra2	90 days	90 days	2 Y	0 Y	-0.0312 R
sra2	180 days	90 days	4 G	0 Y	-0.0380 R
sra2	30 days	180 days	1 R	0 Y	-0.0631 R
sra2	90 days	180 days	2 Y	0 Y	-0.0623 R
sra2	180 days	180 days	4 G	0 Y	-0.0759 R
sra3	30 days	90 days	3 Y	0 Y	0.0024 Y
sra3	90 days	90 days	9 G	0 Y	0.0024 Y
sra3	180 days	90 days	16 G	0 Y	-0.0001 R
sra3	30 days	180 days	3 Y	0 Y	0.0049 Y
sra3	90 days	180 days	9 G	0 Y	0.0048 Y
sra3	180 days	180 days	16 G	0 Y	-0.0003 R
sra9	30 days	90 days	5 G	0 Y	-0.0133 R
sra9	90 days	90 days	10 G	0 Y	-0.0072 R
sra9	180 days	90 days	15 G	0 Y	-0.0083 R
sra9	30 days	180 days	5 G	0 Y	-0.0266 R
sra9	90 days	180 days	10 G	0 Y	-0.0144 R
sra9	180 days	180 days	15 G	0 Y	-0.0166 R
iwa1	30 days	90 days	2.0031 B	-0.0910 R	-0.0435 R
iwa1	90 days	90 days	2.6410 B	-0.2016 R	-0.0709 R
iwa1	180 days	90 days	2.4244 B	-0.3651 R	-0.1309 R
iwa1	30 days	180 days	1.9120 B	-0.1820 Y	-0.0869 R
iwa1	90 days	180 days	2.4394 B	-0.4031 Y	-0.1418 R
iwa1	180 days	180 days	2.0593 B	-0.7302 Y	-0.2618 R
iwa2	30 days	90 days	8.4603 R	-0.7993 R	-0.0863 R
iwa2	90 days	90 days	10.4093 R	-1.6089 R	-0.1339 R
iwa2	180 days	90 days	5.8002 R	-2.7522 R	-0.3218 R
iwa2	30 days	180 days	7.6610 B	-1.5986 Y	-0.1726 R
iwa2	90 days	180 days	8.8004 R	-3.2200 R	-0.2677 R
iwa2	180 days	180 days	3.0480 B	-5.5044 R	-0.6436 B
sra4	1 year	1 year	13 G	0 Y	0.0088 Y
sra5	1 year	1 year	20 B	2 B	0.0826 R
sra6	1 year	1 year	-7 B	-1 R	-0.2406 R

Table 24: Predicted absolute metric values, absolute metrics differences, and relative metric differences for the nautilus project and the associated indicators.

Metric	Linear model Interval length	Prediction interval length	Predicted metric value	Predicted absolute metric difference	Predicted relative metric difference
sra2	30 days	90 days	0 B	0 Y	-0.1701 R
sra2	90 days	90 days	0 B	0 Y	-0.1694 R
sra2	180 days	90 days	0 B	0 Y	-0.1905 R
sra2	30 days	180 days	0 B	0 Y	-0.3402 R
sra2	90 days	180 days	0 B	0 Y	-0.3389 R
sra2	180 days	180 days	0 B	0 Y	-0.3810 R
sra3	30 days	90 days	0 B	0 Y	-0.3483 R
sra3	90 days	90 days	0 B	0 Y	-0.3319 R
sra3	180 days	90 days	0 B	0 Y	-0.4085 R
sra3	30 days	180 days	0 B	0 Y	-0.6966 B
sra3	90 days	180 days	0 B	0 Y	-0.6639 B
sra3	180 days	180 days	0 B	0 Y	-0.8170 B
sra9	30 days	90 days	0 B	0 Y	-0.2570 R
sra9	90 days	90 days	0 B	0 Y	-1.0000 B
sra9	180 days	90 days	1 R	0 Y	-0.2043 R
sra9	30 days	180 days	0 B	0 Y	-0.5141 B
sra9	90 days	180 days	0 B	-1 R	-1.0000 B
sra9	180 days	180 days	1 B	-1 R	-0.4085 R
iwa1	30 days	90 days	0.0000 B	-0.2629 R	-0.1254 R
iwa1	90 days	90 days	0.0000 B	-0.6211 R	-0.0988 R
iwa1	180 days	90 days	0.0000 B	-1.0963 R	-0.0944 R
iwa1	30 days	180 days	0.0000 B	-0.5259 Y	-0.2508 R
iwa1	90 days	180 days	0.0000 B	-1.2423 R	-0.1977 R
iwa1	180 days	180 days	0.0000 B	-2.1927 R	-0.1887 R
iwa2	30 days	90 days	6.1599 R	-0.2681 R	-0.0417 R
iwa2	90 days	90 days	0.0000 B	-0.7634 R	-1.0000 B
iwa2	180 days	90 days	0.0000 B	-1.3575 R	-0.7241 B
iwa2	30 days	180 days	5.8918 B	-0.5362 Y	-0.0834 R
iwa2	90 days	180 days	0.0000 B	-1.5269 Y	-1.0000 B
iwa2	180 days	180 days	0.0000 B	-2.7150 R	-1.0000 B
sra4	1 year	1 year	0 B	0 Y	-1.0000 B
sra5	1 year	1 year	3 R	0 Y	0.0050 R
sra6	1 year	1 year	-3 B	0 Y	-0.1340 R

Table 25: Predicted absolute metric values, absolute metrics differences, and relative metric differences for the httpd 1.3 project and the associated indicators.

6. Conclusion

The contributions of this deliverable may be summarized as follows. We proposed a data-driven methodology that builds max entropy risk indicators. This methodology is especially well-suited when risks clearly increase or decrease with metric values. It is not intended to replace indicators defined by experts, but it may be useful when there is no such guidance. Then, we applied the methodology to community metrics provided by other QualOSS partners using data collected by the FLOSSMetrics project. This practical application allowed us to point flaws in the definition of a category of metrics measuring trends, called slope metrics in this deliverable. To overcome these flaws, we outlined a predictive formulation of risk assessment. The main benefit of this formulation is that it cleanly separates the definition of the quantity to predict and the choice of model to make that prediction. As a consequence, the definition of indicators becomes more natural and independent of the choice of model family. Using the FLOSSMetrics data, we defined risk indicators for absolute metric values, absolute metric differences, and relative metric differences. Finally, we illustrated the computation of all the metrics and indicators presented in this deliverable on a selection of open-source projects. Unfortunately, it appears that linear models used throughout are not accurate enough to predict the evolution of metrics.

We identify several promising research directions to improve our risk indicators. First, we analysed projects with a limited view that should be expanded with more or better metrics. Second, more flexible prediction models should be used. We should take advantage of the many machine learning algorithms and techniques to select and compare prediction models in a principled way. Finally, a more better optimization method to design max-entropy indicators would improve their quality and interpretability.

REFERENCES

1 Herraiz, Israel, A statistical examination of the evolution and properties of libre software, 2008,http://purl.org/net/who/iht/phd

APPENDIX A

The Free/Libre Open Source Software Metrics (FLOSSMetrics) project provides tools to analyse open source projects and gathers the information collected by these tools in the Melquiades database. To perform our data mining analysis, we retrieved from Melquiades the data extracted by the CVSAnalY2 tool. As described in deliverable D3.1 of the FLOSSMetrics project, CVSAnalY2 analyses the source code repository of an open source project and creates a database whose schema is given in Illustration 22.

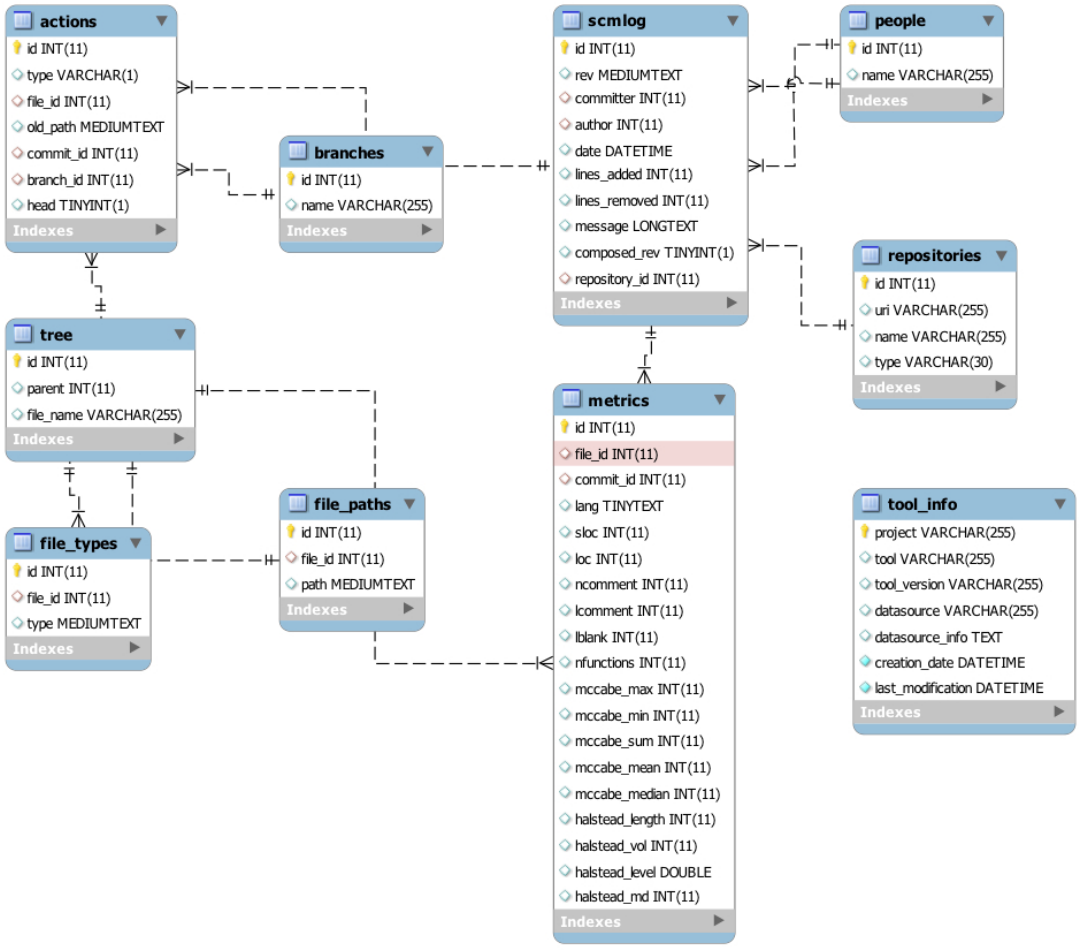

Illustration 22: Schema of the FLOSSMetrics database associated to a project.

Let us briefly describe the parts of this schema that are important to compute the metrics. The scmlog table contains an entry for every commit recorded by the source code management system of the open source project. Among other things, this table measures the author, the number of lines added, the number of lines removed, the message, and the time and date of each commit. A commit consists of one or more actions ('add', 'modify', 'copy' or 'delete') performed on one or more source files. The actions table contains one entry per action performed on a file in a commit. Finally, the tables tree and file_types allows us to identify the location, name, and type (such as 'documentation', 'code', 'image', ...) of each file involved in a commit. On January 13th, 2009, we downloaded from Melquiades the data collected by CVSAnalY2 for 1467 projects.

The metrics and MySQL queries used in this deliverable were designed by other QualOSS partners. We slightly modified some of the queries to allow different time interval lengths. Given a time interval T and a project database, the low-level metrics sra2, sra3, sra9, iwa1, and iwa2 are computed with the following queries.

sra2	SELECT `interval`, COUNT(tbl.committer) AS `cm-sra2` FROM (SELECT s.committer, DATEDIFF(MIN(s.date),(SELECT MIN(s.date) FROM scmlog s)) DIV T AS `interval` FROM scmlog s, actions a, file_types ft WHERE s.id=a.commit_id and a.file_id=ft.file_id and ft.type='code' GROUP BY s.committer) AS tbl GROUP BY `interval` ORDER BY `interval`
sra3	SELECT `interval`, COUNT(tbl.committer) AS `cm-sra3` FROM (SELECT s.committer, DATEDIFF(MIN(s.date),(SELECT MIN(s.date) FROM scmlog s)) DIV T AS `interval` FROM scmlog s, actions a, file_types ft WHERE s.id=a.commit_id and a.file_id=ft.file_id and ft.type<>'code' GROUP BY s.committer) AS tbl GROUP BY `interval` ORDER BY `interval`
sra9	SELECT DATEDIFF(s.date,(SELECT MIN(s.date) FROM scmlog s)) DIV T AS `interval`, COUNT(distinct s.committer) AS `cm-sra9` FROM scmlog s, actions a, file_types ft WHERE s.id=a.commit_id and a.file_id=ft.file_id and ft.type='code' GROUP BY `interval` ORDER BY `interval`
iwa1	SELECT DATEDIFF(s.date,(SELECT MIN(s.date) FROM scmlog s)) DIV T AS `interval`, COUNT(s.id)/COUNT(distinct s.committer) `cm-iwa1` FROM scmlog s GROUP BY `interval` ORDER BY `interval`
iwa2	SELECT DATEDIFF(s.date,(SELECT MIN(s.date) FROM scmlog s)) DIV T AS `interval`, COUNT(s.id)/COUNT(distinct s.committer) `cm-iwa2` FROM scmlog s, actions a, file_types ft WHERE s.id=a.commit_id and a.file_id=ft.file_id and ft.type='code' GROUP BY `interval` ORDER BY `interval`

The metrics sra4, sra5, and sra6 are defined with the notion of core committers. For each project, the core committers are contained in a table core_table built with the following query.

```
SELECT tbaux.myyear,tbaux.committer,tbaux.commits FROM
  (SELECT g.myyear, g.committer, IF(g.committer is null,@sumacu:=0,@sumacu:=@sumacu+g.mycom) AS acusum, IF(g.committer is null,0,g.mycom) AS commits FROM
    (SELECT @sumacu:=0) AS r,
    (SELECT year(s.date) AS myyear, s.committer, COUNT(s.id) AS mycom FROM `PROJECT`.scmlog AS s GROUP BY myyear, s.committer WITH ROLLUP) AS g
     ORDER BY g.myyear, g.mycom
  ) AS tbaux,
  (SELECT YEAR(date) AS myyear, COUNT(id)*20/100 AS nocore FROM `PROJECT`.scmlog GROUP BY myyear) AS tbtotal
 WHERE tbaux.myyear=tbtotal.myyear AND tbaux.acusum>tbtotal.nocore
```

Using the table core_table, we obtain sra4 and sra5 with the next two queries where the time interval length is hard-coded to one year. Recall that sra6 is defined as the difference between sra4 and sra5.

sra4	SELECT Q4.coreyear AS `interval`, COUNT(Q4.committer) AS `cm-sra4` FROM (SELECT MIN(myyear) AS coreyear, committer FROM core_table GROUP BY committer ORDER BY coreyear) AS Q4 GROUP BY Q4.coreyear
sra5	SELECT Q5.coreyear AS `interval`, COUNT(Q5.committer) AS `cm-sra5` FROM (SELECT (y1.myyear+1) AS coreyear, y1.committer FROM (SELECT myyear,committer FROM core_table) AS y1 LEFT JOIN (SELECT myyear, committer FROM core_table) AS y2 ON y2.myyear=y1.myyear+1 AND y1.committer=y2.committer HERE y2.myyear IS NULL AND y2.committer IS NULL AND

	y1.myyear NOT IN (SELECT MAX(myyear) FROM core_table)) AS Q5 GROUP BY Q5.coreyear

Finally, the static metrics sra7, iwa4, iwa5, and iwa7 are computed as follows.

sra7	SELECT SUM(total.sum_months) / COUNT(total.list_committers) AS `cm-sra7` FROM (SELECT new.committer list_committers, COUNT(new.committer) sum_months FROM (SELECT committer, DATEDIFF(date,(SELECT MIN(date) FROM scmlog)) DIV 30 AS `interval` FROM scmlog GROUP BY committer, `interval`) AS new GROUP BY new.committer) AS total
iwa4	SELECT (COUNT(*)/(SELECT COUNT(distinct a.file_id) FROM actions a)) AS `cm-iwa4` FROM (SELECT a.file_id FROM actions a, scmlog s WHERE a.commit_id=s.id GROUP BY a.file_id HAVING COUNT(distinct s.committer)=1 AS g
iwa5	SELECT (SUM(m.sloc)/(SELECT COUNT(distinct committer) FROM scmlog WHERE date>=(SELECT date_sub(MAX(date),interval 1 year) FROM scmlog))) AS `cm-iwa5` FROM metrics AS m
iwa7	SELECT COUNT(distinct a.file_id)/(SELECT COUNT(distinct file_id) FROM actions) AS `cm-iwa7` FROM actions AS a, (SELECT id, committer FROM scmlog s WHERE date>=(SELECT date_sub(MAX(date),interval 1 year) FROM scmlog)) AS g, file_types AS ft WHERE a.commit_id=g.id AND a.file_id=ft.file_id AND ft.type='code'