Reliability Assessment of Service-based Software under Operational Profile Uncertainty

Roberto Pietrantuono[a,*], Peter Popov[b], Stefano Russo[a]

[a] *DIETI, Università degli Studi di Napoli Federico II, Via Claudio 21, 80125 Naples, Italy*
[b] *CSR, City, University of London, Northampton Square, EC1V 0HB, London, UK.*

Abstract

We address the problem of operational reliability assessment through testing of software services delivered on-demand such as Web Services. Software reliability assessment is typically done for a specific *operational profile*: the profile is needed in testing to select or generate test cases (operational testing) in a way statistically similar to the anticipated use of software in operation; the observations of success/failure of test executions are used to predict software reliability in *actual operation*. It is well known that unless the profile is accurate, software reliability predictions obtained via operational testing cannot be trusted.

We present a new way of dealing with the *uncertainty* in the operational profile adopting a two-stage Bayesian inference for reliability assessment. The technique relies on the availability of information about partitions of the input space. The approach is demonstrated on contrived examples and on a case study of real Web Services. We discuss the usefulness of the approach in dealing with two important practical problems: *i)* the true profile in operation differs from the one used in testing, *ii)* the profile in operation is changing continuously.

Keywords: Software reliability, Software testing, Bayes methods, Service-based software, Web service

1. Introduction

The increasing usage of software services (like Web Services) for mission-critical infrastructures – e.g., in the banking [1] and financial domains [2] - as well as for safety-critical systems – e.g., (real-time) embedded systems for domains like industrial automation [3, 4], transportation [5], unmanned vehicles [6, 7], health care [8] - raises the need for accurately assessing their operational reliability. Availability/reliability analysis of service-based software through up-front modeling at design-time is a widely addressed research area [9, 10, 11, 12].

*Corresponding author.
Email addresses: roberto.pietrantuono@unina.it (Roberto Pietrantuono), p.t.popov@city.ac.uk (Peter Popov), stefano.russo@unina.it (Stefano Russo)

OP	operational profile
pdf	probability density function
r.v.	random variable
D	input domain of the service
d	a specific input from D
d_r	an input to the software selected at random from D
$prob(d)$	probability of occurrence of input d
$S \equiv \{S_1, ..., S_n\}$	a decomposition of D into n partitions
$\mathcal{P}_i$	probability that an input is selected from $D \in S_i$ (*probability of partition* S_i)
$OPP \equiv \{\mathcal{P}_1, ..., \mathcal{P}_n\}$	the set of partitions' probabilities (*OP on partitions*)
$prob(d \mid d \in S_i)$ $(i = 1, ..., n)$	conditional probabilities of input d within partition S_i (*operational profile within partitions*)
$\mathcal{F}$	r.v. representing the service probability of failure on a randomly chosen input
$\mathcal{R} \equiv 1 - \mathcal{F}$	r.v. representing the service reliability
$f_{\mathcal{F}}(x)$	*pdf* of the r.v. $\mathcal{F}$
$\mathcal{F}_i$	r.v. conditional probability of failure on partition S_i
$f_{\mathcal{F}_i}(x)$	*pdf* of $\mathcal{F}_i$
$E[\mathcal{F}]$, $E[\mathcal{R}]$, $E[\mathcal{F}_i]$	expected value of r.v. $\mathcal{F}$, $\mathcal{R}$, $\mathcal{F}_i$
$Var(\mathcal{F}_i)$	variance of r.v. $\mathcal{F}_i$
$Covar(\mathcal{F}_i, \mathcal{F}_j)$	covariance of r.v. $\mathcal{F}_i$, $\mathcal{F}_j$
$D(\alpha_1, ..., \alpha_n)$	Dirichlet distribution with parameters $\alpha_1, ..., \alpha_n$
$Beta(a, b)$	Beta distribution of shape parameters a, b
$\Gamma()$	Gamma function
B	Bayes factor

Table 1: Acronyms and notations

However, service providers like Amazon [13] are expected to give assurances (service-level agreements) for failure-free operation that are robust to the uncertainty in the operational profile.

We consider the problem of assessing software services reliability in operation. A key practice in software reliability engineering is the operational profile based testing (or simply *operational testing*) [14], a category of testing techniques shown to suit well reliability assessment [15, 16, 17]. They work by establishing the intended operational profile (OP) of the software under assessment by characterizing how it will be used in operation [18, 19]. The OP is used to generate a suite of test cases; based on observation of successes and failures of test executions, one draws conclusions about its reliability.

It has been argued over the years that unless the profile is captured accurately, the estimates obtained via testing cannot be trusted to give an accurate prediction of reliability in operation [14]. Though acknowledged as important, this issue has been given insufficient attention: operational testing commonly assumes that the OP is accurate. Attempts to construct worst-case bounds on

the impact the OP variation may have on reliability evaluation, e.g. [20] follow the tradition in the critical systems domain of conservative assessment when dealing with the uncertainty of the various affecting factors. Drawbacks of conservative assessment are well known, too. It is clear that accurate reliability assessment typically requires shorter and less expensive verification, to demonstrate that a reliability target is met, than would be required by conservative techniques. For some classes of on-demand software, such as Web applications, the operational profile may even be *unknown in advance*. One can deal with this difficulty defining a space of candidate profiles and assessing the reliability for all of them, establishing the worst that can result. This approach depends crucially on how accurately the space of candidate profiles is captured.

The operational profile may also *vary significantly over time*. Successful applications are subject to continuous improvement, e.g. by adding new functionalities, which may well affect the way software is used. For instance, the way complex Web applications are used changes over time: a profile considered unlikely before software deployment, may turn out to be the norm.

In all cases of unknown, uncertain or variable profile, the service reliability predicted via operational testing may not be accurate. In this paper, we propose a new way of dealing with the *epistemic and aleatory* profile uncertainty in operational testing. Our contributions are: *i)* we propose a Bayesian modeling framework for OP uncertainty and variability; *ii)* based on this, we propose a black-box engineering method exploiting observational data to improve the accuracy of software reliability prediction when its usage profile differs from the estimated one and/or it changes over time; *iii)* we describe through contrived examples how the method can be actioned, and we show its effectiveness experimentally with publicly available Web Services.

2. Related research

The literature on the impact of profile uncertainty on software reliability estimate lacks recent studies. Several early studies in operational testing investigate empirically the sensitivity of reliability to profile uncertainty. They are quite contradictory. The authors in [21, 22] attest a negligible impact of profile error on reliability estimate. Musa [21] experience that in 99.4% of the studied configurations, the estimate error is smaller than the occurrence probability error by a factor greater than 5. Pasquini *et al.* [22] conclude that "*1) the predictive accuracy of the models is not heavily affected by errors in the estimate of the OP; and 2) this relation depends on the accuracy with which the software system has been tested*". More recently, Silva *et al.* [23] claim that the predictive ability of the studied reliability models is not affected by OP variations.

Oppositely, Cai *et al.* [24] experimentally evaluate testing techniques performance for reliability assessment under inaccurate profiles, revealing an influence on the estimate accuracy. Chen *et al.* [25] conduct a case study, using four Software Reliability Growth Models, finding the error of the estimate to grow linearly with the error in the profile estimate.

Such contrasting results depend on the assessment model and on the case study. They are limited to "observing" the effect of the error on the estimate, with true reliability assumed to be known; the OP uncertainty is not encompassed in the modeling technique. In a real situation, the impact of a wrong profile would be revealed only after a long operational time, when the true reliability can be estimated accurately from the failure data.

Besides sensitivity studies, several authors model OP uncertainty explicitly. Originally, Brown *et al.* [26] and Thayer *et al.* [27] propose adjustments to the frequentist estimator of reliability when the OP differs from the testing profile, based on the chi-square measure of their difference. The problem is then addressed by Miller *et al.* [28], who use the Bayesian estimator to estimate reliability when no failure is observed. They also consider the case of the operational and testing profiles being different, and propose three strategies to adjust the estimate: *i)* discarding test cases, *ii)* conducting further test cases, or *iii)* changing the weights in the estimate based on the new profile. Bishop *et al.* [29] focus on deriving conservative bounds for probability of failure on demand, when testing and operational profile differ. Leung [30] presents optimisation models for reliability allocation under an uncertain OP, assuming that the number of functions executed by the customer is geometrically distributed. Recently, Hartman [31] proposes a model capturing the relation between faults and failures and use it to assess the effect of using OPs for reliability growth testing, showing that operational profile is beneficial, but not when a very high reliability is obtained. Building on the work by Whittaker *et al.* [32], Kamavaram and Goseva-Popstojanova [33] use the notion of *entropy* for OP uncertainty: they use discrete time Markov chains to model software architecture and OP, and source entropy to quantify the uncertainty present in such models. In [34] they generalize [33] based on the method of moments. Unlike our approach, these are white-box techniques: they model reliability as a function of software components' reliabilities and of frequencies of control transfer between components. This is done also by Özekici and Soyer [35], who use a continuous-time reliability model and assume a perfect debugging to remove defects revealed by failures.

Cotroneo *et al.* [36] analyse the relation between the profile error and the number of tests required to deliver better reliability than the ideally error-free operational testing. In [17] they propose a Monte Carlo approach to predict the error on reliability estimate caused by a tester-specified maximum profile error. Testing under uncertainty is also addressed by Menghi *et al.*, who deals with problem of creating a credible oracle for cyber-physical systems in the presence of uncertainty that may affect the coverage of an oracle [37]. Zhang *et al.* focus on uncertainty-wise test case generation [38]. Clearly, although both the oracle problem and test case generation are very important in reliability assessment, both are outside the scope of this paper. These can complement to our work, as we take the result of testing to perform the assessment.

Our proposal resumes the early work by Adams [39], who suggested that the Dirichlet distributions can be used to compute confidence intervals of reliability estimates. We extend it to encompass profile variability, and formalize an actionable method exploiting observational data for dynamic model selection

so as to cope with an uncertain or changing OP. Other authors use Dirichlet distributions to support reliability-oriented testing in software systems. Camilli *et al.* concentrate on ways of reducing uncertainty of the parameters characterising the actions of a Markov Decision process for model-based testing [40]. Our focus is on accounting for the impact of the operational profile, which may vary (significantly) over time, and on the probability of failure of a randomly selected input. About model selection, our work exploits the Bayes factor to this aim. The recent work [41] by Vanslette *et al.* proposes a general metric (Bayesian Validation Metric, BVM) to select among alternative models. While our focus is not on model selection *per se*, the BVM is worth exploring in the future for the problem we address in this work.

3. Background and problem statement

Acronyms and notations used throughout the paper are listed in Table 1.

3.1. Software operational profile

Consider a software service, whose inputs are requests made to the service through its API.[1] Generally, the likelihood of selecting an input from D will vary: some inputs are more likely than others. These differences are captured by the *operational profile* (OP) [18], a probabilistic measure, $prob(d)$, that is the probability that input $d \in D$ is submitted to software for processing.

In operational testing, the input space is typically split into n non-overlapping partitions [24], $S=\{S_1, \ldots, S_n\}$ and $S_i \cap S_j = \emptyset \mid i \neq j$. In this case, the operational profile is often defined in two stages:

- A probability distribution is defined on the set of partitions S, which defines the probability $prob(d_r \in S_i)$ – denoted with $\mathcal{P}_i$ - of selecting at random an input d_r from partition S_i:

$$\mathcal{P}_i \equiv prob(d_r \in S_i) = \sum_{d \in S_i} prob(d) \quad (i = 1, \ldots, n). \tag{1}$$

- The conditional probabilities $p(d \mid d \in S_i)$ of selecting input d from within partition S_i can be expressed as:

$$prob(d \mid d \in S_i) = \frac{prob(d)}{\mathcal{P}_i} \quad (i = 1, \ldots, n). \tag{2}$$

Note that the probabilities (1) and (2) are defined over different domains: the former over the set of partitions (we refer to it as *operational profile on partitions*, OPP); the latter over the inputs of a partition (*operational profile within partitions*). We explicitly point out that the n probabilities $\mathcal{P}_i$, summing up to 1 by their nature, have (n-1) degrees of freedom, i.e., (n-1) partition probabilities can be defined so that $\sum_{i=1}^{n-1} \mathcal{P}_i \leq 1$, and the last one is given by $\mathcal{P}_n = 1 - \sum_{i=1}^{n-1} \mathcal{P}_i$.

[1] Selecting an input from the input space D and submitting it to the service corresponds to issue a request to the service through its API; thus, "Input" and "request" are used synonymously in the following.

3.2. Dealing with profile uncertainty

The probabilities (1) and (2) capture the *aleatory uncertainty* about the likelihood of an input being selected at random from the input space. In principle they are estimable with an arbitrary accuracy: it would suffice to observe software in operation for unlimited period of time. If this were possible, then these probabilities will be known with certainty.

Unlimited observations of software in operation cannot be afforded since D is, in general, very large. While with limited observations one might be able to estimate quite accurately the partition probabilities $\mathcal{P}_i$, precisely estimating the conditional probabilities $prob(d \mid d \in S_i)$ for every single input is infeasible. The very idea of partitioning D and having a much smaller number of partitions than that of inputs is motivated by the desire for a coarser model for the OP.

Infeasibility of estimating $prob(d \mid d \in S_i)$ can be dealt with by making additional assumptions. Finding *plausible* assumptions is difficult and instead *convenient* assumptions are often made in practice, which may be incorrect. One such assumption is that all inputs of a partition are *equally likely*. Another one is that conditional probabilities $prob(d \mid d \in S_i)$ are not affected by a change of the likelihoods of partitions. In this paper we adopt the latter assumption, i.e. that the operational profile changes only affect the probabilities of partitions.

Given a limited knowledge about the true OP (e.g., due to limited observations of software in operation), the estimates of probabilities (1) and (2) are subject to epistemic uncertainty. We focus on epistemic uncertainty of the probabilities of partitions (OPP), which are treated as random variables with their corresponding distributions. We apply Bayesian inference to update the epistemic uncertainty about OPP and discuss practical implications.

4. Modeling framework and application scenarios

4.1. Reliability modeling framework

We assume, in line with the literature [42, 43, 24], that reliability is expressed as the probability of not failing on a randomly chosen input $d_r \in D$. Let $\mathcal{F}$ be a random variable (r.v.) that represents this probability. The service reliability then can be expressed via the r.v. $\mathcal{R} = 1 - \mathcal{F}$.

Let $\mathcal{F}_i$ be the r.v. representing the probability of service failure on an input d_r selected from partition S_i. Assuming that the profile on partitions does not affect the likelihood of the inputs within partitions, each conditional probability $\mathcal{F}_i$ is suitably represented as a r.v. with *pdf* $f_{\mathcal{F}_i}(x)$. We assume that Beta distribution with shape parameters a_i, b_i – $Beta(a_i; b_i)$ is appropriate for $\mathcal{F}_i$, since it offers flexibility and simplifies Bayesian inference, as is detailed below. The expected value of each $\mathcal{F}_i$ with Beta distribution is [44]:

$$E[\mathcal{F}_i] = a_i/(a_i + b_i). \tag{3}$$

Consider the case that OPP is known *with certainty*, i.e., the values $\mathcal{P}_1{=}\overline{P_1}$, $\ldots, \mathcal{P}_n{=}\overline{P_n}$ are known constants. In this case, the probability of failure on an

input d_r selected from D according to that profile is a weighted sum of the n conditional $\mathcal{F}_i$, the weights being the known probabilities $\overline{P_1}, \ldots, \overline{P_n}$:

$$\mathcal{F} = \sum_{i=1}^{n} \overline{P_i} \cdot \mathcal{F}_i, \qquad E[\mathcal{F}] = \sum_{i=1}^{n} \overline{P}_i \cdot E[\mathcal{F}_i] \tag{4}$$

Let us further assume that the n conditional $\mathcal{F}_i$ are *independently distributed random variables*. This is a *plausible assumption* in those cases when an assessor is not going to change the belief (i.e., epistemic uncertainty) associated with $\mathcal{F}_i$ if (s)he sees evidence of poor/good conditional probability of failure in some of the other partitions.[2] We observe that the product $\overline{P_i} \cdot \mathcal{F}_i$ in Equation (**??**) is itself a random variable. Denoting with $f_{\mathcal{F}_i}^{P_i}(x)$ its marginal distribution,[3] the *pdf* of $\mathcal{F}_i$ can be expressed as a convolution:

$$f_{\mathcal{F}}(x \mid \mathcal{P}_1 = \overline{P_1}, ..., \mathcal{P}_n = \overline{P_n}) = f_{\mathcal{F}_1}^{P_1}(x) \circledast \cdots \circledast f_{\mathcal{F}_n}^{P_n}(x). \tag{5}$$

We can now remove the assumption that the profile is known with certainty (captured by $\mathcal{P}_1$=$\overline{P_1}$,..., $\mathcal{P}_n$=$\overline{P_n}$). Since partition probabilities are dependent, following [39] we model the epistemic uncertainty about OPP using a *multivariate distribution*, namely the Dirichlet distribution, $D(\alpha_1, ..., \alpha_n)$, with parameters $(\alpha_1, ..., \alpha_n)$ for n variates with (n-1) degrees of freedom, defined by [44]:

$$f_{\mathcal{P}_1,...,\mathcal{P}_n}(p_1, ..., p_n) = \frac{\Gamma(A)}{\prod_{i=1}^{n} \Gamma(\alpha_i)} \left(\prod_{i=1}^{n-1} p_i^{\alpha_i - 1} \right) \left(1 - \sum_{i=1}^{n-1} p_i \right)^{\alpha_n - 1} \tag{6}$$

where $A = \sum_{i=1}^{n} \alpha_i$, and $\Gamma()$ is the Gamma function.

The marginal distribution of each $\mathcal{P}_i$ variate is a Beta distribution with shape parameters (α_i, A-α_i), $Beta(\alpha_i, A\text{-}\alpha_i)$ [9]. The moments of the $\mathcal{P}_i$ variates are given by [44]:

$$E[\mathcal{P}_i] = \frac{\alpha_i}{A}, \tag{7}$$

$$Var(\mathcal{P}_i) = \frac{\alpha_i \cdot (A - \alpha_i)}{A^2 \cdot (1 + A)}, \quad Covar(\mathcal{P}_i, \mathcal{P}_j) = \frac{-\alpha_i \cdot \alpha_j}{A^2 \cdot (1 + A)}, \quad j \neq i. \tag{8}$$

Using the formula of the total probability, Equation (5) becomes:

$$\begin{aligned} f_{\mathcal{F}}(x) &= \int f(x \mid \mathcal{P}_1, ..., \mathcal{P}_n) f_{\mathcal{P}_1,...,\mathcal{P}_n}(p_1, ..., p_n) dp_1 \ldots dp_n \\ &= \int \left[f_{\mathcal{F}_1}^{\mathcal{P}_1}(x) \cdot ... \cdot f_{\mathcal{F}_n}^{\mathcal{P}_n}(x) \right] f_{\mathcal{P}_1,...,\mathcal{P}_n}(p_1, ..., p_n) dp_1 \ldots dp_n. \end{aligned} \tag{9}$$

Equation (9) provides the *marginal distribution of the service probability of failure*, which accounts for the epistemic uncertainty related to the profile and the n conditional probabilities of failure, $\mathcal{F}_i$.

The marginal distribution of $\mathcal{F}$ given by Equation (9) can be used to compute various metrics of interest for the service under study. One can compute the

[2] We acknowledge that treating the conditional probabilities of failure in partitions as independent random variables is a limitation of the current work and plan to extend it for the more general case when one may want to capture dependencies between them. This is further discussed in Section 8, among threats to validity.

[3] This distribution can be trivially derived from $f_{\mathcal{F}_i}(x)$.

expected value (and other moments) of the service probability of failure, hence the expected value of the service reliability $\mathcal{R}$, which are given by:

$$E[\mathcal{F}] = \sum_{i=1}^{n} E[\mathcal{P}_i] \cdot E[\mathcal{F}_i], \qquad E[\mathcal{R}] = 1 - E[\mathcal{F}]. \tag{10}$$

Moreover, one can compute the risk that the true probability of failure can turn out to be *unacceptably high* (i.e. exceed a given threshold). This risk is represented by the *tail of the distribution* of the service probability of failure:

$$prob(\mathcal{F} \geq T) = \int_{T}^{1} f_{\mathcal{F}}(x)dx. \tag{11}$$

Another question of interest is knowing the *likelihood of surviving the next M input requests without a failure*. This can be obtained as:

$$prob(no\ failure\ in\ next\ M\ inputs) = \textstyle\int_0^1 (1-x)^M \cdot f_{\mathcal{F}}(x)\, dx. \tag{12}$$

The above expressions may be computed from $\mathcal{F}$, which in turn depends on the data observed in operation: *i)* the number of inputs processed correctly and incorrectly in partitions – these will be used to update the uncertainty about conditional probabilities of failure in partitions, $f_{\mathcal{F}_i}(x)$; *ii)* the number of inputs selected from each partition, to update the uncertainty about the partition probabilities, captured by $D(\alpha_1, ..., \alpha_n)$.

4.2. Application scenarios

We envisage two important circumstances of interest:

- The operational profile is *fixed*. In this case the epistemic uncertainty about the OP will diminish as more and more observations come from monitoring the service in use. The distributions of the conditional probabilities of failure in partitions ($\mathcal{F}_i$), too, will become narrower and narrower as more observations are collected, and asymptotically their whole mass will be concentrated in a single point. This asymptotic case may require observations much longer than one can afford prior to deployment. Thus, we foresee the method to be useful in the initial period after deployment.
- The operational profile and/or the service itself is *subject to change* (e.g., due to new functionalities, which may affect the way the service is used). In this case, one can monitor the service behavior for possible changes of the OP on partitions and of the $\mathcal{F}_i$, and re-compute the service reliability and other metrics of interest, like those of Equations (11) and (12). The asymptotic case for the stable profile outlined above may be simply *unattainable* due to frequent changes in the case of a variable profile. Hence, one may wish to discard "old" observations, if the current profile differs significantly from the past. For such circumstances we define a procedure to capture the relevance of the observations in judging the operational profile, where recent observations are given a higher weight than those reflecting the profile in the more distant past.

The next two Sections show how to use the framework in both cases.

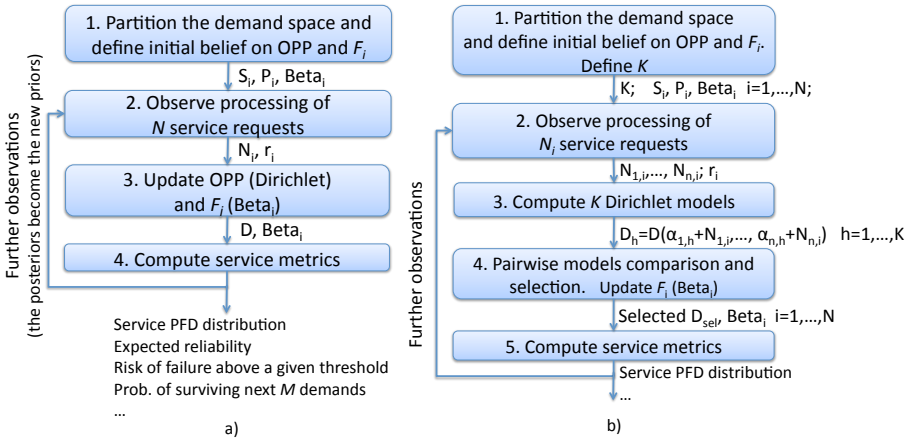


Figure 1: Steps of the method for: a) stable, b) variable operational profile.

5. Stable operational profile

5.1. Method

The method proposed in the scenario of a stable but uncertain OP is shown in Figure 1 a). First, the input space of the service under study is partitioned, and a pre-deployment OP on partitions is defined. Then, the initial beliefs are expressed by defining the initial parameters of thte n-variate Dirichlet distribution – capturing the uncertainty about the OPP - and the n univariate Beta distributions, expressing the uncertainty about the conditional probabilities of failure. In case of "ignorance" (i.e. in the absence of any subjective preferences[4]), the initial beliefs are modeled by a Dirichlet and $Beta_i$ distributions with all parameters equal to 1. When the service is put in operation and a number, N, of input requests are processed, the following information is collected:

- the number of inputs $N_1, ..., N_n$ submitted to partitions $S_1, ..., S_n$, respectively, with $N_1 + ... + N_n = N$;
- the number of failures observed in partitions, $r_1, ..., r_n$, respectively.

The collected information N_i, r_i is used to update the uncertainty related to OPP and the probabilities of failure in partitions, as follows:

- The Dirichlet distribution $D(\alpha_1, ..., \alpha_n)$ modeling the OPP before the new observation, with the new information $N_1, ..., N_n$, will become [44]:

$$D(\alpha_1 + N_1, ..., \alpha_n + N_n). \tag{13}$$

[4] "Ignorance" and "ignorance prior distribution" are part of the jargon used in Bayesian assessment, to indicate that an assessor has no preferences over the range of possible values of a given parameter. Our use of these terms is consistent with the jargon.

- The updated distribution of the conditional probability of failure $\mathcal{F}_i$ in partition S_i, $Beta(a_i, b_i)$, and its expected value, will become:

$$f_{\mathcal{F}_i} = Beta_i(a_i + r_i, b_i + N_i - r_i), \quad E[\mathcal{F}_i] = \frac{a_i + r_i}{a_i + b_i + N_i}. \tag{14}$$

With the new D, $Beta_i$, the $\mathcal{F}$ distribution is given by Equation (9). This allows us to compute the desired service metrics, e.g.: expected reliability; the risk that the probability of failure is above a given threshold (Equation 11); the probability to survive without failures the next M input requests (Equation 12).

5.2. Examples

5.2.1. Example 1

Consider a software service with an input space divided in n=5 partitions, S=$\{S_1, \ldots, S_5\}$, whose operational profile is expected *not to change* in operation, and that this profile is believed to be known with *certainty*.

Assuming no prior knowledge about the occurrence of failures within partitions, the priors $f_{\mathcal{F}_i}(x)$ are set to $Beta_i(a_i$=$1, b_i$=$1)$.

Now suppose the service is deployed and that monitoring it in operation in a time interval (step 2) produces the following observations about the number of requests issued to partitions and the number of failures in partitions:

N_1=300, N_2=800, N_3=1,500, N_4=1,000, N_5=400; N=4,000;
r_1=2, r_2=1, r_3=1, r_4=1, r_5=0.

With the observations, the assessor can update (step 3) the conditional probabilities of failure in partitions and their mean values using the Equations (14):

$f_{\mathcal{F}_1}(x) = Beta_1'(3, 299)$, $f_{\mathcal{F}_2}(x) = Beta_2'(2, 800)$, $f_{\mathcal{F}_3}(x) = Beta_3'(2, 1500)$
$f_{\mathcal{F}_4}(x) = Beta_4(2, 1000)$, $f_{\mathcal{F}_5}(x) = Beta_5'(1, 401)$.

$E[\mathcal{F}_1] = \frac{3}{302}$, $E[\mathcal{F}_2] = \frac{2}{802}$, $E[\mathcal{F}_3] = \frac{2}{1502}$,
$E[\mathcal{F}_4] = \frac{2}{1002}$, $E[\mathcal{F}_5] = \frac{1}{402}$.

We illustrate effect of OPP on the service reliability for 3 different operational profiles: OPP_1, OPP_2 and OPP_3.

Case 1: $OPP_1 = \{P_1 = 0.10,\ P_2 = 0.20,\ P_3 = 0.40,\ P_4 = 0.25,\ P_5 = 0.05\}$.

The reliability assessor can then use Equation (4)[5] to compute the expected value of the service probability of failure $\mathcal{F}$ and the reliability $\mathcal{R}$ (step 4):

$$E[\mathcal{F}] = \sum_{i=1}^{5} P_i \cdot E[\mathcal{F}_i] = 0.002648, \quad E[\mathcal{R}]\text{=1-}E[\mathcal{F}]\text{=0.997352.}$$

Case 2: $OPP_2 = \{P_1 = 0.40,\ P_2 = 0.20,\ P_3 = 0.10,\ P_4 = 0.25,\ P_5 = 0.05\}$.

Under this OPP the least reliable partition, S_1, is significantly more likely, while partition S_3 is significantly less likely than under OPP_1. The expected probability of failure and reliability become noticeably worse:

$$E[\mathcal{F}]\text{=0.005229}, \quad E[\mathcal{R}]\text{=0.994771.}$$

Case 3: $OPP_3 = \{P_i$=$0.20, i$=$1, ..., 5\}$

[5] The P_i are still believed certain at this stage of the example, hence the use of Eq. (5).

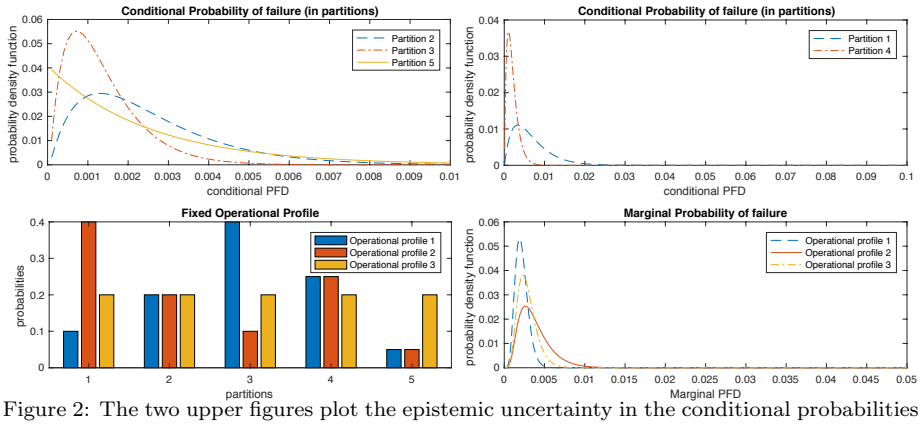


Figure 2: The two upper figures plot the epistemic uncertainty in the conditional probabilities of failure for partitions 3, 4 with higher reliability (left) and for partitions 1, 2, 5 with lower reliability (right). The third plot (bottom left) shows the 3 sample fixed profiles on partitions (OPP_1, OPP_2, OPP_3). The last plot (bottom right) shows the distributions of the marginal probabilities of service failure under the three profiles.

Finally, in case of equally likely partitions, the expected values become:

$$E[\mathcal{F}]=0.003649, \quad E[\mathcal{R}]=0.996351.$$

So far, the initial OPP is believed certain, hence the above expected values ignore the related epistemic uncertainty. The reader may have noticed that the observations indicate significant deviation from the assumed OPP. For instance, under OPP_1 one would expect to see about 200 requests from partition 5, which is significantly fewer than the observed 400 from S_5. However, the uncertainty about the conditional probabilities of failure in partitions, $\mathcal{F}_i$, captured by the Beta distributions, has been taken into account. Under the assumption that these can be treated as independent random variables, the assessor can use Equation (5) to derive the marginal probability of failure for the three profiles. These are shown in Figure 2. Clearly, the operational profile affects not only the expected value of the probability of failure but also the epistemic uncertainty in the distribution of the marginal probability of failure. We note that the OPP in the examples were chosen to be quite different: in OPP_1 and OPP_2 the probability of selecting an input from the partition with the worst reliability, S_1, is swapped with that of the more reliable partition S_3; OPP_3 defines equally likely partitions. The bottom-right plot in Figure 2 shows that the failure distribution under OPP_2 is shifted to the right (higher failure probability) compared to OPP_1 and OPP_3, due to the higher probability of selecting an input from the least reliable partition S_1.

5.2.2. Example 2

Now, let us illustrate how we deal with the uncertainty in OPP, which is captured by the Dirichlet distribution. Assume that the assessor was ignorant before deployment and used the first N=4,000 requests and the counts N_i of requests for partitions to define his/her initial belief about the OPP by modeling

profile uncertainty with the following Dirichlet distribution:

$$D(\alpha_1{=}300,\ \alpha_2{=}800,\ \alpha_3{=}1{,}500,\ \alpha_4{=}1{,}000,\ \alpha_5{=}400).$$

As with Example 1 we assume that the assessor started with ignorance about the probabilities of failure in partitions and then observed the same number of failures in partitions as in Example 1: $r_1{=}2,\quad r_2{=}1,\quad r_3{=}1,\quad r_4{=}1,\quad r_5{=}0.$

The expected values of the $\mathcal{P}_i$ variates are given by Equation (7):

$$E[\mathcal{P}_1] = 0.075,\ E[\mathcal{P}_2] = 0.20,\ E[\mathcal{P}_3] = 0.375,\ E[\mathcal{P}_4] = 0.25,\ E[\mathcal{P}_5] = 0.10.$$

The assessor would then compute (step 4) the marginal probability of failure using Equation (9), accounting for the epistemic uncertainty on both the OPP and the probabilities of failure within partitions, and the new expected service probability of failure and service reliability using Equations (10):

$$E\,[\mathcal{F}]{=}0.002491, \qquad E\,[\mathcal{R}]{=}0.997509.$$

The reliability predicted this way is slightly different from the one estimated before deployment – reflecting the small difference between the profile OPP_1 believed to be certain before deployment and the uncertain profile derived from the actual post-deployment N_i observations - but is more faithful, as the N_i observations in operation (even though limited) corroborate or correct the initial "static" estimate. Figure 3 (bottom-left plot) reports the two distributions of the marginal probability of failure.

Let us now see how the reliability prediction will change as a result of further observations (the iteration in Figure 1). Assuming the assessor observes further $N^{'}{=}100$ requests to the service (hence, the previous posteriors become the new priors), let us consider the following two sets of outcomes.

Observation 1:

$$N_1^{'}{=}8,\quad N_2^{'}{=}22,\quad N_3^{'}{=}39,\quad N_4^{'}{=}23,\quad N_5^{'}{=}8;$$
$$r_1^{'}{=}0,\quad r_2^{'}{=}0,\quad r_3^{'}{=}0,\quad r_4^{'}{=}0,\quad r_5^{'}{=}0.$$

The observed $N_i^{'}$ affect the Dirichlet distribution characterising the uncertainty on the OPP, whose posterior given by expression (13) is the Dirichlet:

$$D^{'}(\alpha_1{=}308,\ \alpha_2{=}822,\ \alpha_3{=}1539,\ \alpha_4{=}1023,\ \alpha_5{=}408).$$

with expected values of the $\mathcal{P}_i$ variates given by Equation (7):

$$E[\mathcal{P}_1]{=}0.0751, E[\mathcal{P}_2]{=}0.2005, E[\mathcal{P}_3]{=}0.3754, E[\mathcal{P}_4]{=}0.2495, E[\mathcal{P}_5]{=}0.0995.$$

Using expressions (14) to obtain the conditional probabilities of failure in partitions and then Equation (10), the new expected values of $\mathcal{F}$ and $\mathcal{R}$ are:

$$E\,[\mathcal{F}]{=}0.002430, \qquad E\,[\mathcal{R}]{=}0.997570.$$

Clearly, as the relative occurrences $N_i^{'}/N^{'}$ are in line with the initial OPP and no failures are observed in the $N^{'}$ requests, the expected reliability has slightly improved.

Observation 2:

$$N_1^{'}{=}10,\quad N_2^{'}{=}45,\quad N_3^{'}{=}30,\quad N_4^{'}{=}8,\quad N_5^{'}{=}7;$$
$$r_1^{'}{=}0,\quad r_2^{'}{=}1,\quad r_3^{'}{=}2,\quad r_4^{'}{=}1,\quad r_5^{'}{=}0.$$

Using expression (13), the posterior Dirichlet distribution will become:

$$D^{'}(\alpha_1{=}310,\ \alpha_2{=}845,\ \alpha_3{=}1530,\ \alpha_4{=}1008,\ \alpha_5{=}407),$$

with expected values of the $\mathcal{P}_i$ variates given by Equation (7):

$$E[\mathcal{P}_1]{=}0.0756, E[\mathcal{P}_2]{=}0.2061, E[\mathcal{P}_3]{=}0.3732, E[\mathcal{P}_4]{=}0.2459, E[\mathcal{P}_5]{=}0.0993.$$

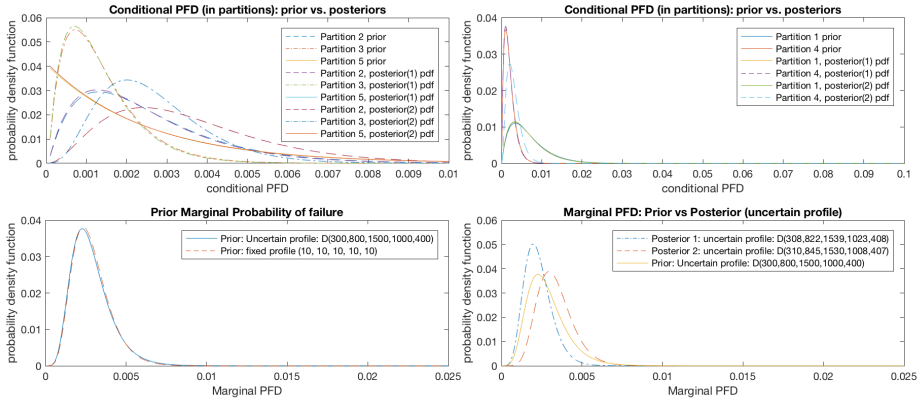


Figure 3: The two upper figures plot the epistemic uncertainty in the conditional probabilities of failure for partitions 2, 3 and 5 with higher reliability (left) and for partitions 1 and 4 with lower reliability (right) before (prior) and after (posterior 1 and posterior 2) observations 1 and 2. The third plot (bottom left) shows the distributions of the marginal probabilities of service failure under a fixed profile and under the uncertain profile modeled by the Dirichlet. The last plot (bottom right) shows the distributions of the marginal probabilities of service failure before (prior) and after (posterior 1 and posterior 2) observations 1 and 2.

The conditional posterior distributions of the probability of failure in partitions will be different than *Observation*1, this affecting the new expected values of $\mathcal{F}$ and $\mathcal{R}$ as follows:

$$E[\mathcal{F}]=0.003404, \qquad E[\mathcal{R}]=0.996596.$$

For both observations, the failure probability, conditional on partitions, and the marginal ones are plotted in Figure 3. It can be seen – from the above computations and from the bottom-right plot of Figure 3 - that the predicted service probabilities of failure (Posterior 1 and 2) are distinctly different from the initial belief (Prior). Unlike *Observation 1*, in *Observation 2* the relative occurrences $N_i^{'}/N^{'}$ are not in line with the initial OPP, and some failures are observed: accordingly, the expected reliability has become worse.

6. Variable operational profile

6.1. Method

In case the OP changes over time, estimates of the service reliability, which are accurate yet promptly reactive to changes, can be made using a range of schemes, depending on how the history of observations in operation is taken into account when updating the Dirichlet distribution. In the case of "small" changes, accounting in the prior for the entire history might be acceptable [39]. In the case of significant and rapid profile changes, discarding the possibly irrelevant history and re-starting with "ignorance" might be preferable. Cases may also be envisaged, whereby accounting in the prior only for the very recent history might be best. Within this range of schemes – from keeping all past observations in the prior to discarding them all - we propose the iterative method shown in Figure 1 b), to chose the prior best suited for the pace of change.

Another option for describing the uncertain OPP might be to use a weighted sum of distributions[6], which account for different lengths of the observed requests history. For example, let us assume that the past profile, $D(\alpha_1, ..., \alpha_n)$, is accounted for with some weighting coefficient, ($0 \leq \mathrm{W} \leq 1$), while the new profile (constructed by starting with "ignorance" and updated with the last N observations), $D(1 + N_1, ..., 1 + N_n)$, is given weight (1-W). The posterior distribution then will be:

$$f_{P_1,...,P_n}(p_1, ..., p_n) = W \cdot D(\alpha_1, ..., \alpha_n) + (1 - W) \cdot D(1 + N_1, ..., 1 + N_n). \quad (15)$$

If this approach is adopted, the attractiveness of the Dirichlet distribution as a prior – leading to a posterior which is also a Dirichlet, the parameters of which can be easily derived from the observations $N_1, ..., N_n$ - will be lost. The *pdf* $f_{\mathcal{P}_1,...,\mathcal{P}_n}(p_1, ..., p_n)$ is no longer guaranteed to be a Dirichlet distribution, and its computation may pose some technical difficulties.

We propose to run several Bayesian models in parallel, and to select for reliability predictions the model which provides *the most accurate prediction of the operational profile at the time a prediction is made.* The candidates are all Dirichlet models, using various history lengths or models defined by Equation (15). To this aim, we divide the history of observations into *iterations.* A candidate model, M_h, will account for the history up to h previous iterations, with $h = 1,...,K$, K being the maximum number of past iterations to consider. With this approach, each candidate prior remains a Dirichlet distribution, allowing for analytic inference (based on the conjugate property of Dirichlet and the multinomial likelihood of the observations). Specifically:

- At iteration i, $N_{1,i},...,N_{n,i}$ requests – with $N_{1,i}+...+N_{n,i}=N_i$ - are observed for partitions $S_1, ..., S_n$, respectively, and the profile is updated.
- K Dirichlet distributions are computed, using the requests observed in the last iteration, and considering the history up to the previous $h = K$ iterations. At iteration i ($i \geq K$), we have K models:

  $$M_h = f_{P_1,...,P_n}(p_1, ..., p_n) = D(\alpha_{1,i-h} + N_{1,i}, \ldots, \alpha_{n,i-h} + N_{n,i}) \quad (16)$$

  where h=1 to K. The parameters $\alpha_{1,i-h},...,\alpha_{n,i-h}$ account for the cumulative number of observations per partition between iterations (i-h) and i. These models consider histories of various lengths, representing the observations more or less well depending on when and to what extent the profile changed.
- These candidate models are then pair-wise compared by means of *posterior odds* on the posterior Dirichlet distributions only. If we are indifferent between the candidate prior beliefs in the OP, using the *Bayes Factor B* is the same as using the *posterior odds*, which is equal to the likelihood ratio. For instance, with two candidate models, M_1 and M_2, for the prior

[6] Or indeed other models, which may be judged as adequate for specific circumstances.

of the operational profiles, the posterior odds can be expressed as:

$$posterior\ odds = \frac{P(M_1|data)}{P(M_2|data)} = \frac{P(data|M_1)}{P(data|M_2)} \cdot \frac{P(M_1)}{P(M_2)} = B \cdot [prior\ odds] \quad (17)$$

where *data* represents the requests $N_{1,i}$,...,$N_{n,i}$ observed in the last iteration. Given the same prior, $prior(M_1)$=$prior(M_2)$, hence *prior odds*= 1, model M_1 is preferred if the Bayes factor is greater than 1, meaning that it describes better the observed data (i.e., how the observed requests are split among partitions).

Comparing models allows addressing a well-known problem in Bayesian inference, often left to intuition, namely how long the history needs to be (i.e., how to choose K) for learning properly. Indeed, this can bring to scalability problems, but the estimate's accuracy *vs* computational cost trade-off is decided by the user depending on the needs/resources. The most expensive choice is to compare, at every iteration, all the models and take the best one. While the least expensive choice is to compare only two models and only at "relevant" iterations. A practical compromise strategy is to compare the model considering only the observations in the last iterations (i.e., without history) against a model with either *a)* all the observations from the beginning ($K = i$), or *b)* the observations up to a given number of past iterations ($K < i$) deemed to be relevant for the problem under study, or *c)* up to known/hypothesised change points of the process (for instance, if the tester has reasons to think that the operational profile has changed – e.g., a new functionality is released - or a change is detected through other techniques).

6.2. Examples

We use contrived examples of variable profile to show how the method selects the model, which best captures the current profile. The examples only deal with the uncertainty on OPP, while the conditional probabilities of failure in partitions are ignored. Let us consider only the models of the two extreme cases: *i)* keeping the full history in the prior since the beginning of the observations (namely, M_h with h=K), and *ii)* completely ignoring the history (M_h with h=1). Consider n=5 partitions, with a uniform profile: p_1=p_2=p_3=p_4=p_5=0.2. Let us assume that the observations are split into iterations of 40 service requests and that the OPP is updated at the end of each iteration. The initial uncertainty in the OPP is captured by a Dirichlet distribution $D(1, 1, 1, 1, 1)$. We further assume that the profile changes after 800 requests (i.e., after iteration 20). We consider two cases for the profile change:

- **Case 1.** The profile change is relatively minor: p_1=0.10, p_2=0.05, p_3=0.30, p_4=0.25, p_5=0.30. Consider the likelihood of the two models M_k, M_1 given the data before and after the change (whose ratio gives the Bayes Factor under the mentioned assumption of equal priors). Figure 4 plots the likelihood values between iterations 0 and 300. We observe that the likelihood values from the models are very small numbers (in the order of 1.0E-10). Yet, it is quite clear, what the ordering between them is: before the profile change the difference is unclear (the ordering of the likelihoods

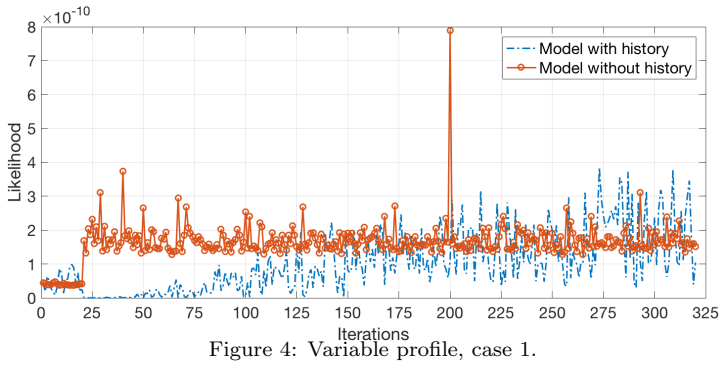


Figure 4: Variable profile, case 1.

oscillates), but afterwards the likelihood of the model without history is clearly greater, which indicates that this model captures better the observations between iterations 20 (when the profile changes) and 150. After iteration 150 the likelihoods computed with the two models become again hardly distinguishable. This behaviour does indicate that after a profile change the model without history provides more accurate predictions.

- **Case 2.** The changed profile is defined as follows: p_1=0.01, p_2=0.60, p_3=0.30, p_4=0.04, p_5=0.05. Clearly, the new profile differs significantly from the uniform profile. Figure 5 plots the likelihood values of the iterations computed with the two models for iterations 0 and 150. Although the pattern after the change is similar to what we observed in the previous example – the model without history reacts to the change better - we see that the difference now is much more clearly pronounced. It is worth noting the gradual improvement of the performance of the history-aware model, but the history-aware model will need much longer to "catch up" with the model in which the history is ignored.

The above examples convey how our model selection method works in the presence of a profile change. Tuning the method parameters (e.g., the maximum

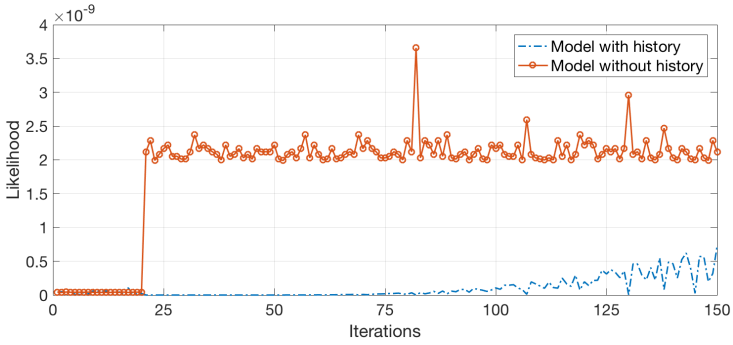


Figure 5: Variable profile, case 2.

history length, or the frequency of models comparison) and how many intermediate models should be included besides the two extreme cases, may affect the velocity the models adapt to a profile change, which in turn affects the choice of model which gives the best predictions.

7. Empirical study

7.1. Subjects

The proposed method is experimentally evaluated on five subjects, namely the third-party Web Services for Natural Language Processing (NLP) listed in Table 2. They are stateless and scalable building blocks for larger systems, offering a REST-like interface – a mainstream form of software services.

Renku is a language detector that processes natural language text and is able of identifying over 100 languages. *Sonnet* is a tokenization engine, which breaks input text into its individual tokens based on rules or trained models – a feature commonly required by NLP applications. *Idyl E3* is an entity extraction engine that infers named-entities from text. *Prose* is an engine which extracts sentences from input text. *Verso* is a text preprocessing engine for common tasks in an NLP pipeline, like removing special characters and stemming.

To derive partitions, the *specification-based partitioning* criterion is adopted [45]. The input arguments of each API method of the Web Service under assessment are grouped in *equivalence classes* based on their type and on the API documentation. We considered both valid (e.g., a string formatted as per documentation) and invalid equivalence classes (e.g., a string with non-printable characters) to assess the service also under more rare circumstances [46]. A *partition* is a combination of equivalence classes, one per input argument.

A partition is selected according to a user-provided testing or operational profile, in the testing and operational phase respectively. Then, a request drawn from the selected partition is a set of input values of an API method picked up randomly from the equivalence classes of that partition.To run a request, the generated inputs are packed in a HTTP request sent to the API method using the conventional GET, PUT, POST and DELETE, in line with the REST paradigm. The HTTP status code of the response is parsed to determine whether a failure has occurred or not. Based on the response, we distinguish:

- *Correct reply*: a *2xx* status code (indicating success) for a request with inputs belonging to valid equivalence classes; or a *4xx* status code (*client*

Service name	**Description**	**# Partitions**
Renku	*Language Detection Engine*	5
Sonnet	*Tokenization Engine*	4
Idyl E3	*Entity Extraction Engine*	6
Prose	*Sentence Extraction Engine*	4
Verso	*Text Pre-processing Engine*	4

Table 2: NLP Web Services selected as experimental subjects.

error) for an incorrect request. These responses are correct replies to incorrect requests, which the client is required to manage.

- *Failure*: the application raises an unexpected unmanaged exception, sent to the client, which is reported as *5xx* status code (server error); or the application provides an *Inconsistent reply*: *i)* a *2xx* reply is obtained when an invalid input is submitted (these are *silent* failures); *ii)* an unexpected *4xx* reply, namely a valid request is issued, but an error is notified.

7.2. Experiment design and procedure

The goal of the experiment is to assess the estimate's accuracy of the probability of failure $\mathcal{F}$ when the true profile in operation differs from the testing profile. Two different scenarios are implemented:

- a *stable* profile case, wherein the OP is kept fixed in the operational phase. This case emulates the situation in which the assessor has reasons to expect a quite stable profile.

- a *variable* profile case, wherein the profile is varied during the operational phase. In this case (or a profile for which we can not make substantiated hypotheses in terms of stability), we use the dynamic model selection.

In both cases the method is compared against a baseline technique, namely the Kalman filters.The experiment targets these research questions:

- **RQ1**: In the stable profile scenario, does the proposed method improve the accuracy of the estimate of $\mathcal{F}$ over the estimate obtained with the testing profile and over the Kalman filter approach?

- **RQ2**: In the variable profile scenario, does the proposed method improve the accuracy of the estimate of $\mathcal{F}$ over the estimate obtained with the testing profile and over the Kalman filter approach?

The reference metric in both cases to check the accuracy is the offset (i.e., the absolute difference) between estimated and true values of the failure probability.

7.2.1. Stable profile

In the case of stable profile, the experimental procedure, given an experimental subject, is as follows:

1. **Generate the testing profile** P_T. The profile is expressed as a vector of values representing the probabilities of selecting a test case from the partitions during testing. We assume that a uniform testing profile is used over partitions (namely, the *OPP* is uniform).
2. **Generate a "true" profile** P. P is generated by changing the testing profile by a random quantity. A variation factor is established, v, to get a modified profile. A random number, $n(i)$, is generated for partition i, by uniformly sampling between $[-v; +v]$. The values of p are then set to: $p(i)=p_r(i)+n(i)$. Two adjustments are made to bind values to the interval

$[0;1]$: first, if $p(i) < 0$ then $p(i) = 0$; second, values are normalized: $p(i) = p(i)/\sum_{j=1}^{m} p(j)$. This generates a true profile that differs from the testing profile by approximately $(v \cdot 100\%)$.[7]

3. **Testing session execution**. A testing session is run according to the conventional OP testing. The testing profile P_T is used to select partitions. The budget is assumed to be T=500 test cases. The number of invocations and of failures observed per operation is recorded.
4. **Post-testing $\mathcal{F}$ computation ($\mathcal{F}_{Test}$)**. After testing, using the number of requests and of failing requests per partition, we get the parameters of the Dirichlet distribution capturing the profile and of the Beta distributions capturing the conditional partition's probabilities of failure $\mathcal{F}_i$. These are put together to get the service probability of failure distribution, $f_{\mathcal{F}}$, according to Equation 9. The distribution allows one to compute various measures of interest such as the expected value of $\mathcal{F}$ or a percentile (e.g., a conservative estimate such as the 90^{th} percentile).
5. **Post-operation $\mathcal{F}$ computation ($\mathcal{F}_{Op}$)**. A number of $K = 5$ iterations, each of $N = 500$ runs, is then executed in order to mimic the operational phase, with input requests selected according to the "true" profile P (i.e., operations are invoked with probability $p(i)$, i=$1, ..., m$). After all iterations, the $\mathcal{F}$ distribution is computed by considering the invocations and failures observed, again using Equation 9 parametrised by the new observations. This allows for updating the estimate w.r.t. the testing phase by considering the executions under the true profile P. Using the same data, the expected probability of failure according to the Kalman filter technique, $\overline{\mathcal{F}}_{KF}$, is also computed (details in Section 7.4).
6. **"True" expected $\mathcal{F}$ computation ($\overline{\mathcal{F}}_{True}$)**. After the operational phase, 10,000 further test cases are run under the true profile P. The frequentist $\mathcal{F}$ is computed: this is assumed to be the "true" expected $\mathcal{F}$, given the very high number of runs compared to $T + N \cdot K$ runs already executed. Post-testing and post-operation expected values of $\mathcal{F}$ are compared against $\overline{\mathcal{F}}_{True}$ to assess the pace of change of the expected $\mathcal{F}$ post-testing and post-operation.

7.2.2. Variable profile

In the case of variable profile, we mimic an operational phase where the true profile P changes at some point. To this aim, steps 5-6 are replaced by:

5' **Post-operation $\mathcal{F}$ computation ($\overline{\mathcal{F}}_{Op}$).** We set again a number of iterations K=5, with N=500 requests per iteration, in which requests are selected according to the true profile P; then, the test goes on for further K=5 iterations (with N=500 requests), in which cases requests

[7] Approximation is given by the first adjustment, which sets to 0 a possible negative value. In the stable profile experiment, v=0.3 (a variation of 30% of the operational with respect to the testing profile). In the variable profile experiment, two values are used, v=0.3 and v=0.7.

are selected according to a new true profile. To assess the impact of the profile change on the provided $\mathcal{F}$ estimate, we run two experiments in which the true profile P changes by a small amount (v=0.3) and by a large amount (v=0.7), respectively. Such modified new profiles are denoted as P_1 and P_2 and are used in experiment 1 and experiment 2, respectively. In both experiments, at the end of all iterations, the following measures are computed: $\overline{\mathcal{F}}_{Op}$ and $\overline{\mathcal{F}}'_{Op}$, as well as the the Kalman filter $\overline{\mathcal{F}}_{KF}$. $\overline{\mathcal{F}}_{Op}$ is the $\mathcal{F}$ computation not considering the model selection described in Section 6 – thus, obtained by the Dirichlet distribution considering all the history like in the stable profile case. $\overline{\mathcal{F}}'_{Op}$ is the $\mathcal{F}$ computation using the model selection approach.

6' The same as step 6, but the true reliability is obtained under the most recent true profile – namely P_1 and P_2 for the two experiments.

For reproducibility, the code and results are made available at:
http://github.com/dessertlab/BayesianReliabilityAssessment

7.3. Evaluation metrics

The following offset metrics are used in the evaluation:

- The absolute difference between $\overline{\mathcal{F}}_{Test}$, the expected $\mathcal{F}$ after testing, and $\overline{\mathcal{F}}_{Op}$, the post-operation one, and the difference between the 90^{th} percentiles of the two estimates. These show how the method updates the $\mathcal{F}$ estimate when the true and testing OP differ.
- The absolute difference between $\overline{\mathcal{F}}_{Test}$ and the true $\mathcal{F}$, $\overline{\mathcal{F}}_{True}$, which we call *post-testing offset*. It represents the error that tester would commit in assessing $\mathcal{F}$ by ignoring the variation of the operational profile at runtime.
- The absolute difference between $\overline{\mathcal{F}}_{Op}$ and $\overline{\mathcal{F}}_{True}$, which we call *post-operation offset*. It represents the error that the tester would commit in assessing $\mathcal{F}$ by using our approach integrating the variation of the OP at runtime with respect to the estimated profile at testing time (in short, *runtime information*).
- The absolute difference between $\overline{\mathcal{F}}'_{Op}$ and $\overline{\mathcal{F}}_{True}$, which we call *post-operation with model selection offset*. It represents the error that tester would commit in assessing $\mathcal{F}$ by using the dynamic model selection.
- The absolute difference between $\overline{\mathcal{F}}_{KF}$ and $\overline{\mathcal{F}}_{True}$, which we call *Kalman filter offset*. It represents the error that the tester would commit in assessing $\mathcal{F}$ by using the Kalman filter technique.

These allow us to evaluate the effectiveness of the proposed method, the benefit of integrating testing-time estimate with runtime data, and the benefit of the dynamic model selection when a profile variation occurs.[8]

[8]The offsets are obtained under one execution of the two scenarios under the specified profile, which gives the distributions of $\mathcal{F}_{Test}$, $\mathcal{F}_{Op}$ and $\mathcal{F}'_{Op}$ (for the Kalman filter, it gives a point estimate). The comparison is between the means of these distributions, as the focus is

7.4. Kalman filter formulation

The KF is a well-known estimation technique used in numerous fields of engineering to estimate the state of a dynamic system (or, generally, an unknown variable of interest), given the measurements collected and the uncertainty inherently associated with the system model and with the measurements process (called, respectively, *process* and *measurement* "noise", both assumed to be zero-mean Gaussian white noise). This latter assumption simplifies the treatment but can also compromise the accuracy whenever is not met in practice.

We have implemented a discrete non-stationary Kalman filter for estimating the expected value of $\mathcal{F}$, denoted as $\overline{\mathcal{F}}_{KF}$, representing the *state* to be known. Measurements about the state in iteration k, denoted as y_k, are the observations of failed requests over the issued requests. The model used for predicting $\overline{\mathcal{F}}_{KF}$ is represented by Eq. 10, thus using the expected values from the Dirichlet and Beta distributions. At each iteration, the "process noise" Q (i.e., the uncertainty associated with the process) is represented by the variance of the updated model – i.e., the variance obtained from using the distributions $D(\alpha_1, \ldots, \alpha_n)$, $Beta_i(a_i, b_i)$ updated by the observations. The "measurements noise" R (i.e., the uncertainty associated with the observations) is represented by the sample variance of $\mathcal{F}$ derived from observations. Both are initialised by the initial model's variance at iteration 1 (with $D(1, 1, \ldots, 1)$, $Beta_i(1, 1)$).

The KF algorithm foresees two steps: prediction and update. During prediction, the state of the system at time k, $\overline{\mathcal{F}}_{KF_k}$, and the variance of the estimation error, E_k, are predicted by means of the system model (*a priori* prediction, denoted as $\overline{\mathcal{F}}^-_{KF_k}$ and E^-_k); the update step aims at fixing the predictions by exploiting collected measurements, giving the *a posteriori* estimates, which are $\overline{\mathcal{F}}_{KF_k}$ and E_k. The usual KF Equations are as follows [47]:

$$
\begin{aligned}
&\textbf{Predict} && \\
&\overline{\mathcal{F}}^-_{KF_k} = A_k \cdot \overline{\mathcal{F}}_{KF_{k-1}} + B_k \cdot u_k && \textit{Project the state ahead} \\
&E^-_k = A_k \cdot E_{k-1} A^T_k + Q_k && \textit{Project the error covariance ahead} \\
&\textbf{Update} && \\
&L_k = E^-_k \cdot C^T_k (C_k \cdot E^-_k \cdot C^T_k + R_k)^{-1} && \textit{Compute the Kalman gain} \\
&\overline{\mathcal{F}}_{KF_k} = \overline{\mathcal{F}}^-_{KF_k} + L_k (y_k - C_k \cdot \overline{\mathcal{F}}^-_{KF_k}) && \textit{Update estimate with measurements} \\
&E_k = (I - L_k \cdot C_k) E^-_k && \textit{Update the error covariance}
\end{aligned}
\tag{18}
$$

where: A_k is the state transition matrix used to predict the state at time k from the state at time $k-1$; $B_k \cdot u_k$ can model a possible system's input (not applicable to our case); C_k models a possible linear relation between the state and the measurement (not applicable to our case); L_k is the innovation gain – a larger value gives more importance to recent measurements compared to past observations (the optimal gain can be computed as $E^-_k C^T_k (C_k E^-_k C^T_k + R_k)^{-1}$

on evaluating how, on average, the methods accurately track the effect of the profile change on the failure probability and converge to the true value. We do not focus here on distribution-based comparison (e.g., by confidence intervals), nor, likewise, on multiple repetitions under different profiles with consequent statistical hypothesis test. These are left to future work.

[48]). Since the estimated state is unidimensional in our design, scalars are used instead of matrices, and variances instead of covariances. We have:

$$A_k = 1 + \frac{\sum_i (\overline{\mathcal{P}}_{i,k-1} \Delta \mathcal{F}_{i,k-1}] + \overline{\mathcal{F}}_{i,k-1} \Delta \mathcal{P}_{i,k-1} + \Delta \mathcal{P}_{i,k-1} \Delta \mathcal{F}_{i,k-1})}{\sum_i \overline{\mathcal{P}}_{i,k-1} \overline{\mathcal{F}}_{i,k-1}};$$
$$B_k = 0; C_k = 1; L_k = E_k^- (E_k^- + R_k)^{-1} \tag{19}$$

where: $\overline{\mathcal{P}}_i$ is the expected probability that a request is selected from partition S_i; $\overline{\mathcal{F}}_i$ is the expected conditional probability of failure on partition S_i; the Δ values are the updates of these expected values obtained by updating the Dirichlet and Beta parameters α_i, a_i, b_i with the observations at iteration k-1.

Unlike our technique, this formulation cannot provide information on the unknown distribution of $\mathcal{F}_{KF}$, yielding a point estimate of the expected value of $\mathcal{F}_{KF}$. This prevents a practitioner from computing measures of interest in terms of risk/confidence in the reliability estimate.

8. Results

8.1. RQ1: Performance under a stable operational profile

Table 3 reports the mean of the post-testing distribution ($\overline{\mathcal{F}}_{Test}$), of the post-operation $\mathcal{F}$ distributions as estimated by our approach ($\overline{\mathcal{F}}_{Op}$) and by the Kalman filter ($\overline{\mathcal{F}}_{KF}$), and the *true* expected value of $\mathcal{F}$ computed after 10,000 tests. Columns 2 and 3 show that the true profile makes the post-testing $\mathcal{F}$ estimate change toward a new estimate after operation, with differences ranging from 0.0483 (for the *Prose* service, from 0.0902 to 0.0419) to 0.053 (for the *Idyl E3* service, from 0.1882 to 0.1829). Depending on how the profile changes and on how often the service fails, the estimate including runtime information changes noticeably. Column 4 reports the $\mathcal{F}$ estimate provided by the Kalman Filter.

Figure 6 plots histograms with the offset values of the post-testing and post-operation failure probability values w.r.t. the true failure probability as well as the offset obtained by using the Kalman filter. Note that the offset post-testing is always larger than that post-operation (where 5 operational cycles are considered): using the runtime information allows for lowering the offset by an order of magnitude, going from an average of 0.0243 to 0.0051 (mean values across subjects of post-testing and post-operation offsets, respectively). It should be noted that the result is obtained under a profile change of approximately 30% (v=0.3, see footnote 7): larger deviations can cause the estimated prior to become even further from the operational reliability. We can see that in only

Service	$\overline{\mathcal{F}}_{Test}$	$\overline{\mathcal{F}}_{Op}$	$\overline{\mathcal{F}}_{KF}$	$\overline{\mathcal{F}}_{True}$
Prose	0.0902	0.0419	0.0482	0.0387
Renku	0.2889	0.2653	0.2370	0.2492
Sonnet	0.0941	0.0848	0.0716	0.0851
Idyl E3	0.1882	0.1829	0.0613	0.1820
Verso	0.2134	0.2033	0.2326	0.1984

Table 3: Stable profile: expected $\mathcal{F}$ for the experimental subjects.

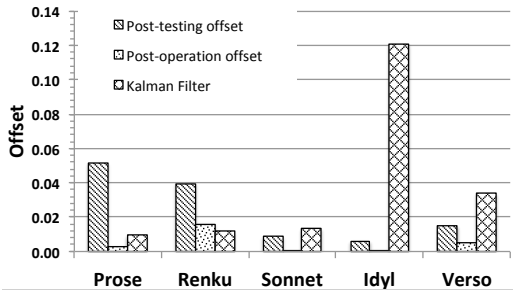


Figure 6: Stable profile: offset values.

one case (*Renku*) the Kalman filter gives an offset value slightly smaller than our approach; in all the other cases, the proposed model outperforms the KFs' estimates. In addition, as stated earlier, our solution provides predictions of distributions rather than a point estimate.

In Table 4, columns 2 and 3 provide the 90^{th} percentiles of the $\mathcal{F}$ distributions at testing and operational time. These are useful in case of interest is a *likely conservative estimate* of the failure probability rather than its expected value. The differences range from 0.0057 to 0.0814. Columns 5 and 6 show the standard deviation of post-testing and post-operation distributions. As in the latter cases more executions are considered, the standard deviation is generally smaller. In summary, our Bayesian assessment provides estimates closer to the true one (Figure 6) and with a reduced uncertainty (Table 4).[9]

Service	$\mathcal{F}_{Test}$ 90^{th} percentile	$\mathcal{F}_{Op}$ 90^{th} percentile	Abs. difference Post-oper. - Post-testing	$\mathcal{F}_{Test}$ standard deviation	$\mathcal{F}_{Op}$ standard deviation
Prose	0.1070	0.0724	0.0346	0.0132	0.0113
Renku	0.3251	0.2848	0.0403	0.0286	0.0156
Sonnet	0.1104	0.0937	0.0167	0.0129	0.0075
Idyl E3	0.2821	0.2764	0.0057	0.0690	0.0683
Verso	0.2344	0.2158	0.0186	0.0169	0.0130

Table 4: Stable profile: 90th percentile and variance of the $\mathcal{F}$.

8.2. RQ2: Performance under a variable operational profile

Tables 5a and 5b report the results of the two experiments with the variable operational profiles (with variation factors v=0.3 and v=0.7, respectively). They report (column $\overline{\mathcal{F}}_{Test}$) the mean of the post-testing $\mathcal{F}$ distribution[10], and of two post-operation $\mathcal{F}$ distributions, corresponding to the situation where *i)* one

[9] As stated, KF produces a point estimate of $\mathcal{F}$; thus, percentiles are not available.

[10] Although, during testing, requests to partitions are taken from the same uniform distribution, there is some difference between post-testing $\mathcal{F}$ values in the two experiments (col. 2 of Table 5a and 5b): because of randomisation (and of the limited number of observations, $T = 500$), the exact number of requests and of failures per partition is different.

Service	$\overline{\mathcal{F}}_{Test}$	$\overline{\mathcal{F}}_{Op}$	$\overline{\mathcal{F}}'_{Op}$	$\overline{\mathcal{F}}_{KF}$	$\overline{\mathcal{F}}_{True}$
Prose	0.0804	0.0324	0.0249	0.0390	0.0245
Renku	0.1794	0.1423	0.1024	0.0357	0.1196
Sonnet	0.0740	0.0821	0.0851	0.0634	0.0847
Idyl E3	0.1622	0.1801	0.1341	0.0182	0.1382
Verso	0.2040	0.2244	0.2941	0.1913	0.2920

(a) Experiment 1 ($v = 0.3$)

Service	$\overline{\mathcal{F}}_{Test}$	$\overline{\mathcal{F}}_{Op}$	$\overline{\mathcal{F}}'_{Op}$	$\overline{\mathcal{F}}_{KF}$	$\overline{\mathcal{F}}_{True}$
Prose	0.0895	0.0486	0.0304	0.0593	0.0289
Renku	0.1804	0.1352	0.0936	0.2163	0.0292
Sonnet	0.0852	0.0853	0.0621	0.0400	0.0683
Idyl E3	0.1940	0.1584	0.1102	0.0073	0.0904
Verso	0.1501	0.1802	0.2402	0.2678	0.2395

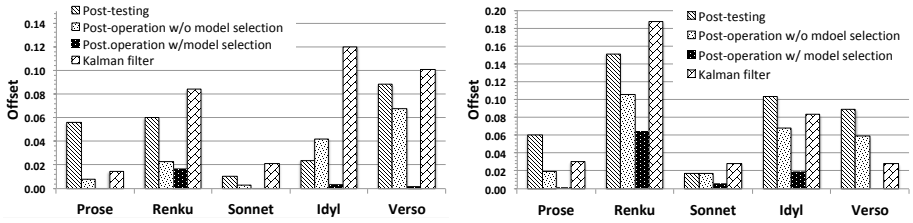
(b) Experiment 1 ($v = 0.7$)

Table 5: Variable profile: expected failure probability. $\overline{\mathcal{F}}_{Op}$: Post-operation without model selection; $\overline{\mathcal{F}}'_{Op}$: Post-operation with model selection; $\overline{\mathcal{F}}_{KF}$: Post-operation with Kalman filter; $\overline{\mathcal{F}}_{True}$: True expected value of $\mathcal{F}$.

always updates the same model *without model selection* (which is the same as in the stable profile case; column $\overline{\mathcal{F}}_{Op}$), and *ii)* one exploits the introduced dynamic model selection strategy to cope with a changing profile (column $\overline{\mathcal{F}}'_{Op}$). The Tables also include the expected $\mathcal{F}$ as estimated by the KF model.

Post-operation distributions are computed after two cycles of K=5 iterations each, under profile P for the first cycle, followed by profile P_1 for the second cycle in experiment 1 (Table 5a, v=0.3), while under P followed by P_2 in experiment 2 (Table 5b, v=0.7). The expectation is that the estimation by the model selection approach ($\overline{\mathcal{F}}'_{Op}$) gets closer to the true reliability compared to the post-testing estimate ($\overline{\mathcal{F}}_{Test}$) as well as to the post-operation case without model selection ($\overline{\mathcal{F}}_{Op}$) and the Kalman filter ($\overline{\mathcal{F}}_{KF}$). Looking at columns 2 to 5 of Tables 5a and 5b, we see that by accounting for runtime information the estimate is updated like in the previous experiment. In all cases, the estimate of the dynamic model selection ($\overline{\mathcal{F}}'_{Op}$) is between the post-testing estimate and the true value (after 5 iterations from the true profile change point, the model is already able to provide better estimates than the post-testing one in all the cases). This happens in all but one case with the approach without model selection, while it happens in just 4 out of 10 cases with the Kalman filter; in the remaining cases, KF would need further iterations for the estimate to converge toward the true value. The difference is more pronounced in the experiment with a more severe profile change (Table 5b): the average absolute differences between the post-operation estimate without (column 3) and with (column 4) model selection is bigger, i.e.: (0.1323–0.1281)=0.00416 in Table 5a vs. (0.1215–0.1073)=0.0142 in Table 5b.

Figures 7a and 7b show the results about the offset with respect to the true value of $\mathcal{F}$. The offset of the post-testing estimate is always bigger than the offset of both post-operation cases. As in the previous experiment, including runtime information leads to considerably lower offset: from an average across the subjects of 0.04768 (average of *Post-testing* offsets across subjects in Fig. 7a) to 0.0285 (average of *Post-operation w/o model selection* offsets in Fig. 7a) and from 0.08434 (average of *Post-testing* offsets of Fig. 7b) to 0.054 (average of *Post-operation w/o model selection* offsets of Fig. 7b). The dynamic model selection makes the improvement bigger, with offsets achieving 0.0049 (average



(a) Experiment 1 ($v = 0.3$) (b) Experiment 2 ($v = 0.7$)

Figure 7: Variable profile: offset values.

of *Post-operation w/ model selection* offsets of Fig. 7a) and 0.01852 (average of *Post-operation w/o model selection* offsets of Fig. 7b).

The offset values of the first experiment are smaller than the second one, because the profile variation is smaller and the model selection approach converges more quickly toward the true value. However, the relative gain is bigger in the second case: the difference between the averages of *post-operation w/ model selection* offsets and *post-testing* offsets is bigger compared to experiment 1 (in the former case, there is an offset reduction of 0.04281 from testing-time to post-operation estimate; in the latter case, the offset reduction is of 0.06582). The dynamic model selection is more decisive to reduce the offset in the cases of stronger profile variations. Comparing the offsets between the true failure probability and the mean failure probability predicted with our approach and with KF, it is clear that KF as a predictor is inferior to the with-model-selection predictor: all offsets computed for $\overline{\mathcal{F}}$, predicted with the KF model, are greater than the offsets computed for the with-model-selection predictor. Indeed, KF does not score very well in comparison with the standard Bayesian inference, i.e. against the without-model-selection predictor, either. KF produces a smaller offset than the without-model-selection predictor only in one case (*Verso*, experiment 2). In summary, these results are consistent with the stable profile case: the with-model-selection predictor performs best, and the improvements increase with the magnitude of the profile variation.

8.3. Threats to validity

Validity of the approach. We have highlighted earlier in the paper the importance of the assumption that the conditional probabilities of failure in partitions *are independently distributed random variables*. The model predictions made with limited observations, the focus of the paper, may be significantly affected by this assumption. Relaxing it and treating the conditional probabilities of failure as *dependent r.v.* is conceptually straightforward and merely requires adding a suitably chosen Copula, to capture the specific knowledge that might be available about software under consideration. Such additional knowledge may come from analysing in detail how software executes requests from different partitions. Such analysis, however, may be infeasible, e.g. if the source code is not available. We are currently *unaware of reliable proxies* which we

could use to capture easily, from observing software in operation, the existence and strength of dependences between conditional probabilities of failure. For this practical reason, the model extension to deal with dependent conditional probabilities of failure is left to future research.

The threats to validity of the results from the empirical study follow.
Internal validity. To generate the true operational profile, a random variation has been applied to the uniform testing profile. Although this could not represent a real profile for the considered services, any generation procedure would be subject to the same threat. The focus of the empirical study was on the ability of our solution to converge to the true reliability by exploiting runtime information and dynamic model selection, regardless what the actual true profile is. The empirical study was conducted with specific parameters settings: v=0.3 and then v=0.7; $T = N = 500$ test cases and requests per iteration; K=5 and then K=10 iterations. Further changing these parameters can impact the speed of convergence of our solution towards the true reliability. Additional threats to internal validity include the correctness of scripts for data collection and of implementation of the experimental code performed by the authors. These have been made available and can be scrutinised by the community.
Construct validity. To define a partition, from which test cases are drawn, we have applied specification-based partition testing, where equivalence classes are defined based on the input arguments of API methods, by inspecting the API documentation. While this the most common practice to partition the input space based on specifications, other criteria could be applied (e.g., structural partitioning). This would lead to different results depending on the overlap between equivalence classes (the more they are overlapped, the more the independence between partitions' conditional probability of failure is violated, the worse the impact on the estimated reliability). Exploring the sensitivity of results to alternative partitioning criteria is left to future research. The true reliability is obtained by running a number of 10,000 requests under the true profile: we assumed this to be sufficient to provide an accurate estimate of true reliability, being it 20 times bigger than the executed test cases T=500, and being the estimator's variance for all services between $1.0E$-5 and $1.0E$-7.
External validity. Results of the empirical study refer to five stateless REST Web Services. They confirm on real subjects the outcome of contrived numerical examples. Care must be taken in extending these results to other programs.

9. Conclusions

We have described a new way of dealing with uncertainty and variability of the operational profile when assessing the reliability of software services delivered on-demand. The technique – based on Bayesian inference - allows a service reliability assessor to deal with: *i)* the deviations of the actual profile from the one used in operational testing before deployment, and *ii)* the changes of the profile over time, due, for instance, to changes in the way the service is used.

We have illustrated the method with contrived examples, showing how software engineers can use it while monitoring the runtime profile changes, to quan-

tify their impact on the reliability estimated through testing before service deployment. We have then presented experiments with real Web Services. The results show the effectiveness of the proposed technique for dealing with the uncertainty in the two scenarios of a stable usage profile, yet different from the one estimated before service delivery, and of a profile which varies over time in the operational phase. For best predictions, the application of the proposed method requires the service input space to be split into non-overlapping partitions.

The method is well suited for web services employed in critical applications in which the operational profile may change over time. While in traditional safety critical applications the profile is typically assumed fixed a new range of critical application becomes increasingly important in which the profile is highly dynamic. A prominent example are applications and services in autonomous cars, which have attracted significant interest by industry and researchers. Weather and driving conditions may change rapidly, thus making the conditions for object recognition and route planning vary dramatically over short periods of time.

Acknowledgment

The work by Pietrantuono and Russo is funded by the EU Horizon 2020 programme under the Marie Skłodowska-Curie grant agreement No 871342. The work by P. Popov is funded in part by the UK GCHQ (grant agreement 4196242) and the AQUAS project, EU ECSEL JU (grant agreement 737475).

References

[1] M. Mazzara, N. Dragoni, A. Bucchiarone, A. Giaretta, S. T. Larsen, S. Dustdar, Microservices: Migration of a mission critical system, IEEE Trans. on Services Computing (2018) 1–1.

[2] A. F. Services, Building mission-critical financial services applications on aws, Tech. rep., Amazon (2019).

[3] H. Bohn, A. Bobek, F. Golatowski, Sirena - service infrastructure for real-time embedded networked devices: A service oriented framework for different domains, in: ICN/ICONS/MCL, 2006, pp. 43–43.

[4] G. Moritz, S. Pruter, D. Timmermann, F. Golatowski, Web services on deeply embedded devices with real-time processing, in: Int. Conf. on Emerging Technologies and Factory Automation, 2008, pp. 432–435.

[5] G. Gehlen, G. Mavromatis, A web service based middleware for mobile vehicular applications, in: Proc. 2nd Int. Workshop on Intelligent Transportation, 2005, pp. 35–39.

[6] D. Rodrigues et al., Application of SOA in Safety-Critical Embedded Systems, in: G. Lee, D. Howard, D. Slezak (Eds.), Convergence and Hybrid Information Technology, Springer, Berlin, Heidelberg, 2011, pp. 345–354.

[7] D. Rodrigues et al., Using SOA in Critical-Embedded Systems, in: 2011 Int. Conf. on Internet of Things and 4th Int. Conf. on Cyber, Physical and Social Computing, IEEE, 2011, pp. 733–738.

[8] A. Al-Humam, Service-oriented architectures for safety-critical systems, Ph.D. thesis (2015).

[9] Y. Zhang, M. Lyu, QoS Prediction in Cloud and Service Computing, SpringerBriefs in Computer Science, Springer, Singapore, 2017.

[10] E. L. Følstad, B. E. Helvik, The cost for meeting sla dependability requirements; implications for customers and providers, Reliability Engineering & System Safety 145 (2016) 136–146.

[11] Z. Zheng, K. S. Trivedi, K. Qiu, R. Xia, Semi-markov models of composite web services for their performance, reliability and bottlenecks, IEEE Trans. on Services Computing 10 (3) (2017) 448–460.

[12] S. Bosse, M. Splieth, K. Turowski, Multi-objective optimization of it service availability and costs, Reliability Engineering & System Safety 147 (2016) 142–155.

[13] C. Newcombe, T. Rath, F. Zhang, B. Munteanu, M. Brooker, M. Deardeuff, How Amazon web services uses formal methods, Comm. ACM 58 (2015) 66–73.

[14] M. R. Lyu (Ed.), Handbook of software reliability engineering, McGraw-Hill, Inc., Hightstown, NJ, USA, 1996.

[15] K.-Y. Cai, Y.-C. Li, K. Liu, Optimal and adaptive testing for software reliability assessment, Information and Software Technology 46 (15) (2004) 989–1000.

[16] J. Lv, B.-B. Yin, K.-Y. Cai, Estimating confidence interval of software reliability with adaptive testing strategy, Journal of Systems and Software 97 (2014) 192–206.

[17] D. Cotroneo, R. Pietrantuono, S. Russo, RELAI testing: A technique to assess and improve software reliability, IEEE Trans. on Software Engineering 42 (5) (2016) 452–475.

[18] J. Musa, Operational profiles in software-reliability engineering, IEEE Software 10 (2) (1993) 14–32.

[19] C. Smidts, C. Mutha, M. Rodríguez, M. J. Gerber, Software testing with an operational profile: OP definition, ACM Computing Surveys 46 (3) (2014) 39:1–39:39.

[20] P. Bishop, R. Bloomfield, Worst case reliability prediction based on a prior estimate of residual defects, in: 13th Int. Symp. on Software Reliability Engineering, IEEE, 2002.

[21] J. Musa, Sensitivity of field failure intensity to operational profile errors, in: 5th Int. Symp. on Software Reliability Engineering, 1994, pp. 334–337.

[22] A. Pasquini, A. Crespo, P. Matrella, Sensitivity of reliability-growth models to operational profile errors vs. testing accuracy [software testing], IEEE Trans. on Reliability 45 (4) (1996) 531–540.

[23] O. Silva, A. Crespo, M. Chaim, M. Jino, Sensitivity of two coverage-based software reliability models to variations in the operational profile, in: Fourth Int. Conf. on Secure Software Integration and Reliability Improvement, 2010, pp. 113–120.

[24] K.-Y. Cai, C.-H. Jiang, H. Hu, C.-G. Bai, An experimental study of adaptive testing for software reliability assessment, Journal of Systems and Software 81 (8) (2008) 1406–1429.

[25] M.-H. Chen, A. Mathur, V. Rego, A case study to investigate sensitivity of reliability estimates to errors in operational profile, in: 5th Int. Symp. on Software Reliability Engineering, IEEE, 1994, pp. 276–281.

[26] J. R. Brown, M. Lipow, Testing for software reliability, SIGPLAN Not. 10 (6) (1975) 518–527.

[27] T. Thayer, M. Lipow, E. Nelson, Software Reliability: A study of large project reality, Vol. 2 of TRW Series of Software Technology, North-Holland, New York, 1978.

[28] K. W. Miller et al., Estimating the probability of failure when testing reveals no failures, IEEE Trans. on Software Engineering 18 (1) (1992) 33–43.

[29] P. Bishop, A. Povyakalo, Deriving a frequentist conservative confidence bound for probability of failure per demand for systems with different operational and test profiles, Reliability Engineering & System Safety 158 (2017) 246–253.

[30] Y.-W. Leung, Software reliability allocation under an uncertain operational profile, Journal of the Operational Research Society 48 (4) (1997) 401–411.

[31] H. Hartmann, A statistical analysis of operational profile driven testing, in: Int. Conf. on Software Quality, Reliability and Security Companion, IEEE, 2016, pp. 109–116.

[32] J. Whittaker, J. Poor, Markov analysis of software specifications, ACM Trans. on Software Engineering and Methodology 2 (1) (1993) 93–106.

[33] S. Kamavaram, K. Goseva-Popstojanova, Entropy as a measure of uncertainty in software reliability, in: 13th Int. Symp. on Software Reliability Engineering, IEEE, 2002, pp. 209–218.

[34] K. Goseva-Popstojanova, S. Kamavaram, Software reliability estimation under certainty: generalization of the method of moments, in: Int. Symp. on High Assurance Systems Engineering, IEEE, 2004, pp. 209–218.

[35] S. Özekici, R. Soyer, Reliability of software with an operational profile, European Journal of Operational Research 149 (2003) 459–474.

[36] D. Cotroneo, R. Pietrantuono, S. Russo, Combining operational and debug testing for improving reliability, IEEE Trans. on Reliability 62 (2) (2013) 408–423.

[37] C. Menghi, S. Nejati, K. Gaaloul, L. C. Briand, Generating automated and online test oracles for simulink models with continuous and uncertain behaviors, in: Proc. 27th ACM ESEC/FSE 2019, ACM, 2019, pp. 27–38.

[38] M. Zhang, S. Ali, T. Yue, Uncertainty-wise test case generation and minimization for cyber-physical systems, Journal of Systems and Software 153 (2019) 1 – 21.

[39] T. Adams, Total variance approach to software reliability estimation, IEEE Trans. on Software Engineering 22 (9) (1996) 687–688.

[40] M. Camilli, C. Bellettini, A. Gargantini, P. Scandurra, Online model-based testing under uncertainty, in: 29th Int. Symp. on Software Reliability Engineering, IEEE, 2018, pp. 36–46.

[41] K. Vanslette, T. Tohme, K. Youcef-Toumi, A general model validation and testing tool, Reliability Engineering & System Safety 195 (2020) 106684.

[42] J. Lv, B.-B. Yin, K.-Y. Cai, On the asymptotic behavior of adaptive testing strategy for software reliability assessment, IEEE Trans. on Software Engineering 40 (4) (2014) 396–412.

[43] P. Frankl, D. Hamlet, B. Littlewood, L. Strigini, Evaluating testing methods by delivered reliability, IEEE Trans. on Software Engineering 24 (8) (1998) 586–601.

[44] I. Albert, J.-B. Denis, Dirichlet and multinomial distributions: properties and uses in Jags, Rapport technique 2012-5, INRA (2012).

[45] M. Young, M. Pezze, Software Testing and Analysis: Process, Principles and Techniques, John Wiley & Sons, 2005.

[46] N. Laranjeiro, M. Vieira, H. Madeira, A robustness testing approach for SOAP Web services, Journal of Internet Services and Applications 3 (2) (2012) 215–232.

[47] R. E. Kalman, A New Approach to Linear Filtering and Prediction Problems, Journal of Basic Engineering 82 (1) (1960) 35–45.

[48] An Introduction to the Kalman Filter, SIC-CRAPH, 2001 Course Notes.