dipartimentoinformatica
Università degli Studi dell'Aquila

Role and impact of error propagation in software architecture reliability

V. Cortellessa and V. Grassi

Technical Report TRCS 007/2006

Dipartimento di Informatica
Università degli Studi dell'Aquila
Via Vetoio Loc. Coppito
I-67010 L'Aquila, Italy

http://www.di.univaq.it

Technical Report TRCS Series

Role and impact of error propagation in software architecture reliability: closed-form solutions

Vittorio Cortellessa [‡] Vincenzo Grassi [†]

[‡] Dipartimento di Informatica

Universita' dell'Aquila

Via Vetoio, 1 - Coppito, 67010, L'Aquila

tel.: (+39) 0862433165, fax: (+39) 0862433131

email: cortelle@di.univaq.it

[†] Dipartimento di Informatica Sistemi e Produzione

Universita' di Roma "Torvergata"

Via del Politecnico 1 - 00133, Roma

tel.: (+39) 0672597380, fax: (+39) 0672597460

email: vgrassi@info.uniroma2.it

Abstract

We present a novel approach to the analysis of the reliability of a software architecture that takes into account an important architectural attribute, namely the error propagation probability. This is the probability that an error, arising somewhere in the architecture, propagates to other components, possibly up to the output. Although this attribute is often neglected in modeling the architecture reliability, it may heavily affect decisions on crucial architectural choices. With our approach, we are able to derive closed-form expressions for the overall system reliability and its sensitivity to variations in the reliability properties of each component (i.e. the probability of failure and the probability of error propagation). This latter result is useful to drive several significant tasks, such as: placing error detection and recovery mechanisms, focusing the design and implementation efforts on critical components, devising cost-effective testing strategies.

Keywords: software architecture, component-based systems, reliability, closed-form solution, state-based model.

1 Introduction

The architectural analysis of software systems aims at characterizing relevant properties (both functional and extra-functional) of the systems in terms of the properties of their "components" and the way the latter are connected and interact [2, 3]. This kind of analysis is raising an increasing interest, because it supports emerging paradigms of software development, like component-based software engineering, COTS-based software development and product line engineering. Moreover, it also supports software evolution, as it can be used for "what if" experiments to predict the impact of architectural changes needed to adapt the system to new or changing requirements.

In this paper, we focus on the architectural analysis of the reliability of a software application, defined as a measure of its ability to successfully carry out its own task. We provide an analytic model for the reliability estimation that, similarly to other approaches, basically relies on information concerning the failure probability and the operational profile of each component.

The novelty of our work, that distinguishes it from most of the existing analytic approaches to architecture-based reliability modeling, consists in the introduction of an important architectural aspect in our model. We embed the error propagation characteristic of each component, that is its capability of propagating (rather than masking) to other components an erroneous value it receives as input from other components. Neglecting this aspect generally corresponds to implicitly assuming a "perfect" propagation, in that any error that arises in a component always propagates up to the application outputs. This may lead, at the best, to overly pessimistic predictions of the system reliability, that could cause unnecessary design and implementation efforts to improve it. Worse yet, if reliability analysis is used to drive the selection of components, it could lead to wrong estimates of the reliability of different component assemblies, thus causing the selection of an assembly which is actually less reliable than others.

Besides an "error propagation aware" reliability model, we also provide an analytic model for the evaluation of the sensitivity of the software architecture reliability to variations in the internal failure and the error propagation characteristics of its components. This latter result provides useful insights for system design, development and testing. Indeed, it can be used to drive the placement of error detection and recovery mechanisms in the system. Moreover, from the viewpoint of project and resource management, it can be used to convey consistent project resources on the most critical components (i.e. the ones with the highest sensitivity). It can also be used to focus the testing efforts on those components where a small change in the failure characteristics may lead to considerable variations in the overall system reliability.

The paper is organized as follows. In section 2 we review the related work. In section 3 we describe the architectural model and the related failure model that we consider. In section 4 we present our approach to the architectural reliability analysis and provide a closed-form solution for the reliability estimation that takes into account the error propagation probabilities. Based on the results of section 4, we provide in section

5 closed-form solutions for the sensitivity of the system reliability with respect to the reliability properties of each component, namely the probability of failure and the probability of error propagation. In section 6 we apply our results to an example and show the relevance of the newly introduced parameters. Finally, we draw some conclusions and give hints for future work in section 7.

2 Related work

Several approaches to the architectural analysis of reliability of modular software systems have been presented in the past. A thorough review can be found in [8], where also issues related to model parameter estimation are discussed. Additional approaches can be found in [6, 18].

According to the classification proposed in [8], most of the architectural reliability models can be categorized as: (i) *path-based models*, if the reliability of an assembly of components is calculated starting from the reliability of possible component execution paths; (ii) *state-based models*, if probabilistic control flow graphs are used to model the usage patterns of components.

These two types of models are conceptually quite similar. One of the main differences between them emerges when the control flow graph of the application contains loops. State-based models analytically account for the infinite number of paths that might exist due to loops. Path-based models require instead an explicit enumeration of the considered paths; hence, to avoid an infinite enumeration, the number of paths is restricted in some way, for example to the ones observed experimentally during the testing phase or by limiting the depth traversal of each path. In this respect, we adopt here a state-based model.

An important architectural attribute that should be taken into account in the reliability analysis of a system is the probability for an error that arises within a component to propagate to other components. Indeed, neglecting this aspect may lead to inexact reliability estimates and wrong choices about the most suitable software architecture. None of the existing architecture-based analytical models considers the impact of error propagation on the estimation of the overall system reliability.

An important advantage of architectural analysis of reliability is the possibility of studying the sensitivity of the system reliability to the reliability of each component. As said in the Introduction, this study can help to identify the critical components which have the largest impact on system reliability, and this information either can be used to focus design and implementation efforts on these components, or may suggest modifications of the system architecture.

Although these advantages are widely recognized [7, 15, 22], a method for computing the sensitivity of the system reliability with respect to each component reliability has been developed only for the model presented in [4]. However, this model does not take into account the error propagation attribute.

An important issue for any analytic model is the estimation of the model parameters. In the case of state-based or path-based models for reliability evaluation, these parameters include the interaction probabilities among the system components, and the probabilistic characterization of each component reliability. Moreover, if we want to take into account the error propagation, we also need to estimate a probabilistic characterization of this additional feature of each component.

Methods for the estimation of the probability of failure of each component and the interaction probabilities have been reviewed in [8], and are extensively discussed in [7]. Regarding the probability of failure, most of the existing models (including ours) assume a single figure of merit for each component (e.g. a failure probability or a time-to-failure distribution), and provide its characterization by averaging over all the possible component usages. Actually, it can be argued that the probability of failure of a component may depend on the distribution of its input values, since they could lead to stress different execution paths within it [11, 16]. The introduction of an input-dependent internal failure model of the component reliability would require the substitution of the single figure of merit with a set of input-dependent figures. Moreover, it would also require, for each component, the definition of a (probabilistic) mapping from input to output domain. This would likely cause an increase in both the model complexity and parameter estimation effort. This increase may or may not lead to any significant improvement in the overall reliability estimation. An example of component reliability model where the availability of this additional information is assumed can be found in [9], where error propagation is not considered. However, input-dependent reliability models deserve further investigation to get insights about the best trade-off between model refinement and model complexity.

Regarding the estimation of the interaction probabilities, besides the approaches reviewed in [8] and [7], a more recent method is discussed in [19], where a Hidden Markov model is used to cope with the imperfect knowledge about the component behavior.

Finally, relevant work on the estimation of the error propagation probability has been recently presented in [1, 12]. In [12] an error permeability parameter of a software module is defined, which is a measure providing insights on the error propagation characteristics of the module. A methodology is devised to extract this parameter from an existing software system through fault injection. Besides, an analytical upper bound is provided, which depends on the number of inputs and number of outputs of the module. In [1] an analytical formula for the estimate of the error propagation probability between two components is provided. This estimate derives from an entropy-based metrics, which basically depends on the sizes of state spaces of interacting components and frequencies of their interactions. Another approach based on fault injection to estimate the error propagation characteristics of a software system during testing was presented in [21].

With respect to the existing literature, the original contributions of this paper can be summarized as follows:

- We define a state-based architectural model for the analysis of reliability, where the error propagation factor is taken into account.

- We derive from this model a closed-form expression for the evaluation of reliability.
- We derive closed-form expressions for the evaluation of reliability sensitivity with respect to the error propagation and the failure probability of each component.

Regarding the estimation of the parameters in our analytic model, we refer to the literature reviewed above. In particular, for the estimate of error propagation characteristics of each component in our model, we point out that it can be faced in two ways: (i) at the architectural level (and, in general, before system deployment) both the analytical upper bound on the error permeability and the entropy-based metrics represent very suitable hints [1, 12]; (ii) upon system deployment, accurate monitoring of the error propagation can be accomplished by means, for example, of fault injection techniques [12, 21].

3 Architectural model

Architecture-based approaches are built on the standard software engineering concept of module. Although there is no universally accepted definition, a module is conceived as a logically independent component of the system which performs a well-defined function. This implies that a module can be designed, implemented, and tested independently. We use the terms module and component interchangeably in this paper. The granularity level adopted to identify the components of an application depends on a tradeoff between the number of components, their complexity and the available information about each component. Indeed, too many small components could lead to a large state space which may pose difficulties in the measurements and parametrization of the model. On the other hand, too few components may lead to loose the distinction of how different components contribute to the system failures.

In our architectural model, we abstractly consider an application consisting of C interacting components. In this model, each interaction corresponds to a control transfer from a component i to a component j, that also involves the transfer to j of some data produced by i. In [20] the distinction between data flow and control flow is discussed with respect to the integration of module and system performance. In our approach, we do not distinguish the two aspects, and we assume that data errors always propagate through the control flow [7].

We assume that the operational profile of each component i is known, and it follows the Markov property. Hence, it is expressed by the probabilities $p(i, j)$ $(1 \leq i, j \leq C)$ that, given an interaction originating from i, it is addressed to component j [8, 23]. It holds the obvious constraint that $(\forall i) \sum_j p(i, j) = 1$.

This abstract model may correspond to different types of modular systems (e.g. component-based software systems, embedded systems made of software and hardware components, merely hardware systems) where the interactions among components are based on messaging, method invocation with parameter passing, or even hardware signals. As an example, according to this model, a method invocation addressed by i

to j is modeled by a pair of interactions where the transferred data correspond, respectively, to the method parameters produced by i and the method results produced by j.

The application ability of producing correct results with respect to specifications is affected by the occurrence of failures during its execution. A failure may arise within a components because of a fault in the software code that implements it. Hence we assume that, besides its operational profile, each component i is also characterized by its internal failure probability $intf(i)$. $intf(i)$ is the probability that, given a correct input, a failure occurs during the execution of i causing the production of an erroneous output. In this definition, "erroneous" refers to the input/output specification of i taken in isolation. $intf(i)$ can be interpreted as the probability of component failure per demand [8].

However, the occurrence of a failure within a component does not necessarily affect the application ability of producing correct results. Indeed, according to [14], a failure arising within a component is only an internal error for the whole application that may or may not manifest itself as an application failure. The former event happens only when the error propagates through other components up to the output of the overall assembly of components. As an extreme example, a component that for any received stimulus produces the same constant value as output completely masks to the other components any received erroneous input. To take into account this effect, we introduce in our failure model the additional parameter $ep(i)$, that denotes the error propagation probability of component i, that is the probability that i propagates to its output a received erroneous input.

We assume that both the internal failure and the error propagation probabilities are independent of the past history of the component.

To conclude the presentation of our model, we point out that, actually, each component could offer a set of different *services*, each service characterized by its own operational profile. This means that, if we denote by i_h $(1 \leq i_h \leq S_i)$ one of the S_i services offered by component i, the operational profile of i would actually consists of a set of probabilities $p(i_h, j_k)$ $(1 \leq j_k \leq Sj)$. $p(i_h, j_k)$ represents the probability that an interaction originating from service i_h is addressed to service j_k offered by component j. From an "abstract" modeling viewpoint, this apparently refined representation does not introduce any relevant modification to the model we have defined, except the substitution of the term "component" with "service". What substantially changes is the practical implication of this representation, that can highly increase the number of parameters to be estimated, making it hardly usable even if theoretically more precise. In this respect our model, based on the characterization of components rather than services, can be considered as the result of a suitable "aggregation" of operational and failure characteristics of component services, thus increasing the model tractability at the cost of some loss in accuracy.

4 An analytic model for reliability embedding error propagation

The operational profile of a component-based software application is expressed by a matrix $\mathbf{P} = [p(i,j)]$, $(0 \leq i,j \leq C+1)$, where each entry $p(i,j)$ represents the probability that component i, during its execution, transfers the control to component j. The rows 0 and $C+1$ of $\mathbf{P}$ correspond to two "fictitious" components that represent, respectively, the entry point and the exit point of the application [23]. These components allow to easily model: (i) the stochastic uncertainty among application entry points, by means of $p(0,j)$ probabilities $(0 \leq j \leq C)$, (ii) the completion of the application by means of the first control transfer to component $C+1$.

Given this model, the application dynamics corresponds to a discrete time Markov process with state transition probability matrix $\mathbf{P}$, where the process state i represents the execution of the component i, and state $C+1$ is an absorbing state. Figure 1 depicts the structure of $\mathbf{P}$, where $\mathbf{Q}$ is a $(C+1)\cdot(C+1)$ sub-stochastic matrix (with at least one row sum < 1), and $\mathbf{c}$ is a column vector with $C+1$ entries.

Hence, the entries of the k-step transition probability matrix $\mathbf{P}^k = [p^{(k)}(i,j)]$ of this process represent the probability that, after exactly k control transfers, the executing component is j, given that the execution started with component i. We recall that $\mathbf{P}^k$ is recursively defined as $\mathbf{P}^0 = I$ (the identity matrix) and $\mathbf{P}^k = \mathbf{P}\cdot\mathbf{P}^{k-1} (k \geq 1)$. Figure 1 also depicts the structure of $\mathbf{P}^k$, where $\mathbf{c}^{(k)}$ is a column vector.

$$\mathbf{P} = \left[\begin{array}{c|c} \mathbf{Q} & \mathbf{c} \\ \hline 0 \;\; 0 \;\; \dots \;\; 0 & 1 \end{array}\right] \qquad \mathbf{P}^{\mathbf{k}} = \left[\begin{array}{c|c} \mathbf{Q}^{\mathbf{k}} & \mathbf{c}^{(\mathbf{k})} \\ \hline 0 \;\; 0 \;\; \dots \;\; 0 & 1 \end{array}\right]$$

Figure 1: The structures of $\mathbf{P}$ and $\mathbf{P}^k$ matrices.

Let us denote by Rel the application reliability, that is the probability that the application completes its execution and produces a correct output. In order to model Rel we introduce the following probabilities in addition to $intf(i)$ and $ep(i)$ defined in section 2 ([1]):

- $err(i)$: probability that the application completes its execution producing an erroneous output, given that the execution started at component i $(0 \leq i \leq C)$;
- $err^{(k)}(i,j)$: probability that the execution reaches component j after exactly k $(k \geq 0)$ control transfers and j produces an erroneous output, given that the execution started at component i $(0 \leq i,j \leq C)$.

Using these definitions we can write the following equations:

[1]By definition, we assume $intf(0) = intf(C+1) = 0$ and $ep(0) = ep(C+1) = 1$.

$$err(i) = \sum_{k=0}^{\infty} \sum_{h=0}^{C} err^{(k)}(i,h)p(h,C+1) \tag{1}$$

$$Rel = 1 - err(0) \tag{2}$$

Equation (1) derives from the assumption that the application completion is represented by the first occurrence of a transition to state $C+1$. The following theorem states a recurrence relation for $err^{(k)}(i,j)$ thus providing, according to equations (1) and (2), the basis for the evaluation of the application reliability.

Theorem 1.

$$err^{(k)}(i,j) = p^{(k)}(i,j) \cdot intf(j) + ep(j) \cdot (1 - intf(j)) \cdot \sum_{h=0}^{C} err^{(k-1)}(i,h)p(h,j) \tag{3}$$

where we assume $err^{(k)}(i,j) = 0$ for $k < 0$ ([2]).

Proof. See Appendix.

For computational purposes, it is convenient to put equations (1) and (3) in matrix form. To this end, let us define the following $(C+1) \cdot (C+1)$ matrices:

- $\mathbf{E}^{(k)} = [err^{(k)}(i,j)](0 \leq i,j \leq C)$;
- $\mathbf{F} = [f(i,j)](0 \leq i,j \leq C)$, a diagonal matrix with $f(i,i) = intf(i)$, and $f(i,j) = 0$ $(\forall i \neq j)$;
- $\mathbf{R} = [r(i,j)](0 \leq i,j \leq C)$, a diagonal matrix with $r(i,i) = ep(i)$, and $r(i,j) = 0$ $(\forall i \neq j)$;

and the following $(C+1)$ column vector:

- $\mathbf{e} = [err(i)](0 \leq i \leq C)$.

Using these definitions, we can rewrite equations (1) and (3) as follows, where $\mathbf{c}$, $\mathbf{Q}$, $\mathbf{Q}^k$ are the vectors and matrices depicted in Figure 1:

$$\mathbf{e} = \sum_{k=0}^{\infty} \mathbf{E}^{(k)} \cdot \mathbf{c} \tag{4}$$

$$\mathbf{E}^{(k)} = \mathbf{Q}^k \cdot \mathbf{F} + \mathbf{E}^{(k-1)} \cdot \mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}) \tag{5}$$

The following theorem provides an explicit form for $\mathbf{E}^{(k)}$.

[2] Note that this basic assumption corresponds to $err^{(0)}(i,j) = intf(j)$ $(\forall i = j)$ and $err^{(0)}(i,j) = 0$ $(\forall i \neq j)$, namely the probability of an erroneous output from component j without any transfer of control is its own probability of *internal failure*.

Theorem 2.

$$\mathbf{E}^{(k)} = \sum_{i=0}^{k} \mathbf{Q}^{k-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^{i} \tag{6}$$

Proof. See Appendix.

It is known that for any substochastic transient square matrix $\mathbf{A}$ it holds $\sum_{k=0}^{\infty} \mathbf{A}^k = (\mathbf{I} - \mathbf{A})^{-1}$ [5]. Now, we observe that, besides $\mathbf{Q}$, also $\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F})$ is a substochastic transient square matrix since, by definition, both $\mathbf{F}$ and $\mathbf{R}$ are diagonal matrices whose diagonal entries are within 0 and 1. Using this observation, we can prove from (4) and Theorem 2 the following theorem, which provides a closed-form expression for $\mathbf{e}$.

Theorem 3.

$$\mathbf{e} = (\mathbf{I} - \mathbf{Q})^{-1} \cdot \mathbf{F} \cdot (\mathbf{I} - \mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^{-1} \cdot \mathbf{c} \tag{7}$$

Proof. See Appendix.

Finally, from equations (2) and (7) we get a closed-form for the application reliability.

5 Sensitivity Analysis

In this section we show how the analytic model developed in section 4 can be used to analyze the sensitivity of the system reliability with respect to the internal failure and error propagation probabilities of its components. For this purpose, let us define the following notations:

- $de\_err(i;l)$: the partial derivative of $err(i)$ with respect to $ep(l)$ $(1 \leq i, l \leq C)$;
- $de\_err^{(k)}(ij;l)$: the partial derivative of $err^{(k)}(ij)$ with respect to $ep(l)$ $(1 \leq i, j, l \leq C)$;
- $di\_err(i;l)$: the partial derivative of $err(i)$ with respect to $intf(l)$ $(1 \leq i, l \leq C)$;
- $di\_err^{(k)}(ij;l)$: the partial derivative of $err^{(k)}(ij)$ with respect to $intf(l)$ $(1 \leq i, j, l \leq C)$;

5.1 Closed-form solution for $de\_err(i;l)$

From equation (2), the sensitivity of the application reliability with respect to $ep(l)$ can be expressed as:

$$\frac{\partial}{\partial ep(l)} Rel = -\frac{\partial}{\partial ep(l)} err(0) = -de\_err(0;l) \tag{8}$$

Hence, our goal is to calculate $de\_err(0;l)$. By differentiating equations (1) and (3) with respect to $ep(l)$ we get, respectively:

$$de\_err(i;l) = \sum_{k=0}^{\infty}\sum_{h=0}^{C} de\_err^{(k)}(ih;l)p(h,C+1) \tag{9}$$

$$\begin{aligned} de\_err^{(k)}(ij;l) &= ep(j)(1-intf(j))\sum_{h=0}^{C} de\_err^{(k-1)}(ih;l)p(h,j) \\ &+ I\{j=l\}(1-intf(l))\sum_{h=0}^{C} err^{(k-1)}(i,h)p(h,l) \end{aligned} \tag{10}$$

where it is assumed that $de\_err^{(0)}(ij;l) = 0$, and $I\{e\}$ is the indicator function defined as: $I\{e\} = 1$ if condition e is true, 0 otherwise.

Analogously to section 4, we derive matrix form expressions from equations (9) and (10). Since, according to equation (8), we are interested in the derivative of $err(0)$, we define the following vector and matrix:

- $\hat{\mathbf{e}} = [\hat{e}(l)]$, $(0 \leq l \leq C)$, with $\hat{e}(l) = de\_err(0;l)$,
- $\hat{\mathbf{E}}^{(\mathbf{k})} = [\hat{e}^{(k)}(lj)]$, $(0 \leq l, j \leq C)$, with $\hat{e}^{(k)}(lj) = de\_err^{(k)}(0j;l)$.

Then, from expression (9) we get:

$$\hat{\mathbf{e}} = \sum_{k=0}^{\infty} \hat{\mathbf{E}}^{(\mathbf{k})}\mathbf{c} \tag{11}$$

whereas from expression (10) we get the following recurrence equation for $\hat{\mathbf{E}}^{(\mathbf{k})}$:

$$\hat{\mathbf{E}}^{(k)} = \hat{\mathbf{E}}^{(k-1)}\mathbf{QR}(\mathbf{I}-\mathbf{F}) + \mathbf{D}^{(k-1)}(\mathbf{I}-\mathbf{F}) \tag{12}$$

where $\hat{\mathbf{E}}^{(0)} = 0$, and $\mathbf{D}^{(k)}$ is a diagonal matrix defined as $\mathbf{D}^{(\mathbf{k})} = [d^{(k)}(lj)]$, $(0 \leq l, j \leq C)$, with:

$$d^{(k)}(jj) = \textstyle\sum_{h=0}^{C} err^{(k)}(0,h)p(hj)$$

$$d^{(k)}(lj) = 0, \forall l \neq j$$

In the following theorem we derive a closed-form matrix expression for $\hat{\mathbf{E}}^{(k)}$.

Theorem 4.

$$\hat{\mathbf{E}}^{(k)} = \sum_{i=0}^{k-1} \mathbf{D}^{(k-i-1)}(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^{i} \tag{13}$$

Proof.

The proof is by induction. For $k=0$ the theorem holds true, since the sum in the right hand side of (13) is empty, and by definition $\hat{\mathbf{E}}^{(0)} = \mathbf{0}$. Let the theorem holds true for $k-1$; we have:

$$
\begin{aligned}
\hat{\mathbf{E}}^{(k)} &= \\
&= \hat{\mathbf{E}}^{(k-1)}\mathbf{QR}(\mathbf{I}-\mathbf{F}) + \mathbf{D}^{(k-1)}(\mathbf{I}-\mathbf{F}) \\
&= (\textit{by induction hypothesis}) \\
&\quad (\sum_{i=0}^{k-2}\mathbf{D}^{(k-i-2)}(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i)(\mathbf{QR}(\mathbf{I}-\mathbf{F})) + \mathbf{D}^{(k-1)}(\mathbf{I}-\mathbf{F}) \\
&= \sum_{i=1}^{k-1}\mathbf{D}^{(k-i-1)}(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i + \mathbf{D}^{(k-1)}(\mathbf{I}-\mathbf{F}) \\
&= \sum_{i=0}^{k-1}\mathbf{D}^{(k-i-1)}(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i
\end{aligned}
$$

qed

Given the result of Theorem 4, we are now able to derive a closed-form matrix expression for $\hat{\mathbf{e}}$.

Theorem 5.

$$
\hat{\mathbf{e}} = \mathbf{D}(\mathbf{I}-\mathbf{F})(\mathbf{I}-\mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1}\mathbf{c} \tag{14}
$$

where $\mathbf{D} = \sum_{k=0}^{\infty}\mathbf{D}^{(k)} = [d(lj)]$ $(0 \le l, j \le C)$ is a diagonal matrix obtained as follows: given the matrix $\mathbf{N} = [n(lj)]$, $\mathbf{N} = (\mathbf{I}-\mathbf{Q})^{-1}\mathbf{F}(\mathbf{I}-\mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1}\mathbf{Q}$, then

$$
d(jj) = n(0j)
$$
$$
d(lj) = 0, \forall l \neq j
$$

Proof.

Looking at (11), let us first elaborate on $\hat{\mathbf{E}} = \sum_{k=0}^{\infty}\hat{\mathbf{E}}^{(k)}$. We have from Theorem 4:

$$
\begin{aligned}
\hat{\mathbf{E}} &= \sum_{k=0}^{\infty}\hat{\mathbf{E}}^{(k)} = \sum_{k=1}^{\infty}\hat{\mathbf{E}}^{(k)} \\
&= \sum_{k=1}^{\infty}\sum_{i=0}^{k-1}\mathbf{D}^{(k-i-1)}(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i \\
&= \sum_{i=0}^{\infty}\sum_{k=i+1}^{\infty}\mathbf{D}^{(k-i-1)}(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{i=0}^{\infty}(\sum_{k=0}^{\infty}\mathbf{D}^{(k)})(\mathbf{I}-\mathbf{F})(\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i \\
&= (\sum_{k=0}^{\infty}\mathbf{D}^{(k)})(\mathbf{I}-\mathbf{F})(\mathbf{I}-\mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1} \\
&= \mathbf{D}(\mathbf{I}-\mathbf{F})(\mathbf{I}-\mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1}
\end{aligned}
$$

The definition of $\mathbf{D}$ stated in this theorem follows from the following considerations: (i) $\mathbf{D}^{(k)}$, by its definition given above, is a diagonal matrix built from the entries of row 0 of the matrix $\mathbf{E}^{(k)}\mathbf{Q}$, and (ii) it is easy to extrapolate from expressions (4) and (7) that $\sum_{k=0}^{\infty}\mathbf{E}^{(k)} = (\mathbf{I}-\mathbf{Q})^{-1}\mathbf{F}(\mathbf{I}-\mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1}$. Finally, the expression for $\hat{\mathbf{e}}$ given in the theorem follows from (11). **qed**

5.2 Closed-form solution for $di\_err(i;l)$

The derivation of the sensitivity with respect to the internal failure probability follows very closely the procedure adopted in the previous section for the sensitivity with respect to the error propagation probability. Hence, we report the theorem proofs of this section in Appendix.

From equation (2), the sensitivity of the application reliability with respect to $intf(l)$ can be expressed as:

$$\frac{\partial}{\partial intf(l)}Rel = -\frac{\partial}{\partial intf(l)}err(0) = -di\_err(0;l) \tag{15}$$

By differentiating equations (1) and (3) with respect to $intf(l)$ we get, respectively:

$$
\begin{aligned}
di\_err(i;l) &= \sum_{k=0}^{\infty}\sum_{h=0}^{C} di\_err^{(k)}(ih;l)p(h,C+1) && (16)\\
di\_err^{(0)}(ij;l) &= I\{i=j \wedge j=l\} && (17)\\
di\_err^{(k)}(ij;l) &= ep(j)(1-intf(j))\sum_{h=0}^{C} di\_err^{(k-1)}(ih;l)p(h,j) \\
&+ I\{j=l\}(p^{(k)}(il)-ep(l)\sum_{h=0}^{C} err^{(k-1)}(i,h)p(h,l)) && (18)
\end{aligned}
$$

According to equation (15), we are interested in the derivative of $err(0)$. Hence, in order to put expressions (16), (17) and (18) in matrix form, we define the following vector and matrix:

- $\tilde{\mathbf{e}} = [\tilde{e}(l)]$, $(0 \le l \le C)$, with $\tilde{e}(l) = di\_err(0;l)$,
- $\tilde{\mathbf{E}}^{(\mathbf{k})} = [\tilde{e}^{(k)}(lj)]$, $(0 \le l, j \le C)$, with $\tilde{e}^{(k)}(lj) = di\_err^{(k)}(0j;l)$.

Then, from expression (16) we get:

$$\tilde{\mathbf{e}} = \sum_{k=0}^{\infty} \tilde{\mathbf{E}}^{(\mathbf{k})} \mathbf{c} \tag{19}$$

whereas from expressions (17) and (18) we get the following recurrence equation for $\tilde{\mathbf{E}}^{(\mathbf{k})}$:

$$\tilde{\mathbf{E}}^{(0)} = \mathbf{A}^{(0)}$$
$$\tilde{\mathbf{E}}^{(k)} = \tilde{\mathbf{E}}^{(k-1)} \mathbf{Q}\mathbf{R}(\mathbf{I} - \mathbf{F}) + \mathbf{A}^{(k)} - \mathbf{D}^{(k-1)}\mathbf{R}$$

where $\mathbf{D}^{(k)}$ is the diagonal matrix defined in section 5.1, and $\mathbf{A}^{(k)}$ is a diagonal matrix defined as $\mathbf{A}^{(k)} = [a^{(k)}(lj)]$ $(0 \le l, j \le C)$ with:

$$a^{(k)}(jj) = p^{(k)}(0j)$$
$$a^{(k)}(lj) = 0, \forall l \ne j$$

In the following theorem we derive a closed-form matrix expression for $\tilde{\mathbf{E}}^{(k)}$.

Theorem 6.

$$\tilde{\mathbf{E}}^{(k)} = \sum_{i=0}^{k} \mathbf{A}^{(k-i)} (\mathbf{Q}\mathbf{R}(\mathbf{I} - \mathbf{F}))^i - \sum_{i=0}^{k-1} \mathbf{D}^{(k-i-1)} \mathbf{R} (\mathbf{Q}\mathbf{R}(\mathbf{I} - \mathbf{F}))^i \tag{20}$$

Proof. See Appendix.

Given the result of Theorem 6, we are now able to derive a closed-form matrix expression for $\tilde{\mathbf{e}}$.

Theorem 7.

$$\tilde{\mathbf{e}} = (\mathbf{A} - \mathbf{D}\mathbf{R})(\mathbf{I} - \mathbf{Q}\mathbf{R}(\mathbf{I} - \mathbf{F}))^{-1}\mathbf{c} \tag{21}$$

where $\mathbf{D}$ is the diagonal matrix defined in Theorem 5, and $\mathbf{A} = \sum_{k=0}^{\infty} \mathbf{A}^{(k)} = [a(lj)]$ $(0 \le l, j \le C)$ is a diagonal matrix obtained as follows:
given the matrix $\mathbf{M} = [m(ij)]$, $\mathbf{M} = (\mathbf{I} - \mathbf{Q})^{-1}$, then

$$a(jj) = m(0j)$$
$$a(lj) = 0, \forall l \ne j$$

Proof. See Appendix.

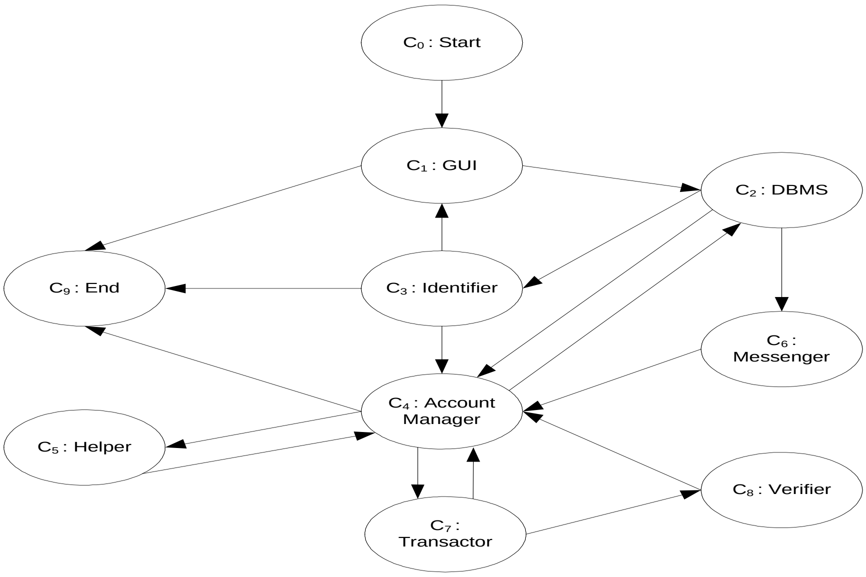

Figure 2: The ATM architecture.

To conclude this section we point out that each entry of $\hat{\mathbf{e}}$ and $\tilde{\mathbf{e}}$ represents, respectively, the sensitivity of the system reliability with respect to the error propagation probability and the internal failure probability of a specific component. Hence, by calculating these vectors, we obtain the reliability sensitivity with respect to the error propagation and internal failure probability of all the system components.

6 Results and analyses

We have used an ATM bank system example to validate our reliability model, taken from [23] and illustrated in Figure 2 ([3]). Shortly, a GUI is in charge of triggering the identification task that is carried out through the interaction of the Identifier and a DBMS. Then the control goes to the Account Manager that is the core of the system. The latter, by interacting with all the other components (i.e. Messenger, Helper, Transactor and Verifier) and by using the data in the DBMS, manages all the operations required from the user during an ATM working session. Without the fictitious *Start* and *End* components, the system is made of components C_1 through C_8.

We devise two sets of experiments on this architecture to show the application of our model. In the first set, based on the closed-form derived in section 4, we compare the reliability prediction we get by neglecting the error propagation impact with the prediction we get when this impact is taken into consideration. In all the experiments, the reliability values where error propagation is neglected have been obtained by setting to 1 the error propagation probabilities of all components. We call this setting as "perfect" propagation. In the second set of experiments, based on the closed-forms derived in section 5, we analyze the sensitivity of the

[3]The system architecture differs from the one used in [23] only by one component that we have not duplicated as we do not consider fault-tolerance in our approach.

	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	$intf(i)$
	Q									**c**	
C_0	0	1	0	0	0	0	0	0	0	0	0
C_1	0	0	0.999	0	0	0	0	0	0	0.001	0.018
C_2	0	0	0	0.227	0.669	0	0.104	0	0	0	0.035
C_3	0	0.048	0	0	0.951	0	0	0	0	0.001	0
C_4	0	0	0.4239	0	0	0.1	0	0.4149	0	0.0612	0.004
C_5	0	0	0	0	1	0	0	0	0	0	0.01
C_6	0	0	0	0	1	0	0	0	0	0	0
C_7	0	0	0	0	0.01	0	0	0	0.99	0	0
C_8	0	0	0	0	1	0	0	0	0	0	0.1001
C_9	0	0	0	0	0	0	0	0	0	1	0

Table 1: Initial values of model parameters.

system reliability to the error propagation and internal failure probabilities of its components.

6.1 Role of the error propagation: experimental results

Table 1 shows the values we have considered for the model parameters, that match the ones used in [23]. In Table 2 the value Rel of the system reliability obtained from equations (2) and (7) is reported (in the bottom row) while varying the error propagation probability $ep(i)$ of all the components (in the top row). In this experiment we have assumed the same $ep()$ value for all components. The first column (i.e. $ep(i) = 1.0$) represents a perfect propagation setting, where each component propagates to its output all the errors that it receives as input. This corresponds to the setting adopted in the reliability models of other authors where error propagation is not considered. For sake of giving evidence to the role of "non-perfect" error propagation, we decrease the $ep(i)$ value by the same quantity (i.e. 0.1) for all the components from the second column on.

$\forall i, ep(i)$	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
Rel	0.4745	0.8261	0.8989	0.9399	0.9494	0.9617	0.9710	0.9784	0.9848	0.9906

Table 2: System reliability vs component error propagation.

It is easy to observe that the perfect propagation assumption brings to heavily underestimate the system reliability. In fact, a decrease of only 10% of the error propagation probability of each component (i.e. from 1.0 to 0.9) suffices to bring a 74% increase in the whole system reliability (i.e. from 0.4745 to 0.8261).

In order to show the negative effects of such underestimation on the system assembly, let us assume that an alternative component $C_{4.1}$ is available for the Account Manager that is functionally equivalent to the one adopted in the initial configuration. The failure characteristics of this component are: $intf(4.1) = 0.008$, $ep(4.1) = 0.9$. If we adopt a failure model that does not take into consideration the error propagation, then the new component cannot apparently improve the system reliability, as its internal failure probability doubles the one of the original component. In fact, the system reliability with the new value of $intf(4.1) = 0.008$ will be $Rel = 0.4594$, which corresponds to an apparent net decrease of 3% with respect to $Rel = 0.4745$ obtained with C_4. This induces a system designer to not considering the new component in the system assembly. But if we adopt our model that embeds the error propagation probability, then the system reliability with the new values of $intf(4.1) = 0.008$ and $ep(4.1) = 0.9$ will be $Rel = 0.7094$, which corresponds to a net increase of 49%. The model, in this case, brings to the evidence at the system level the better error propagation characteristic of the new component. Thus, although $C_{4.1}$ worsen the internal failure probability of C_4, its role of core component in the system architecture (i.e. due to the operational profile the majority of system execution paths traverse this component) brings to prefer the former that propagates the errors with a lower probability.

6.2 Sensitivity analysis: experimental results

In order to exploit the closed form expressions that we have found for the derivatives of system reliability, we start from the derivative values in the initial parameter setting of Table 1 that brought to $Rel = 0.4745$, as shown in the previous section. These values are summarized in Table 3 along with the original component parameters, with the exception of C_0 and C_9 that are fictitious components.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
$ep(i)$	1	1	1	1	1	1	1	1
$\frac{\partial Rel}{\partial ep(i)}$	-0.0199	-1.7830	-0.4360	-4.2732	-0.4246	-0.2001	-1.6031	-1.5853
$intf(i)$	0.018	0.035	0	0.004	0.01	0	0	0.1001
$\frac{\partial Rel}{\partial intf(i)}$	-0.5051	-2.1502	-0.4705	-3.8948	-0.3864	-0.2159	-1.4442	-1.5870

Table 3: System reliability derivatives for the initial values of model parameters.

From Table 3 it straightforwardly appears that the failure characteristics of certain components (i.e. probability of internal failure and error propagation probability) affect the system reliability much more severely than the characteristics of other components. In particular, quite high absolute values for both derivatives are obtained for components C_2, C_4, C_7 and C_8. This means that variations in their failure characteristics would affect the system reliability more than variations in the characteristics of the remaining components.

The set of critical components that have emerged is not surprising, as from the transition matrix $\mathbf{Q}$ in Table 1 they appear to be the most visited components. In other words, the majority of the system execution paths traverse these components, therefore a change in their failure characteristics would affect the majority of the system results.

These data, that can be easily obtained from our closed-form expressions, can be very relevant to support decisions during the system development. For example, a system developer may decide to concentrate the efforts on critical components to improve their failure characteristics. In fact, small gains in those components would lead to large gains in the whole system reliability.

We support with a numerical example this consideration.

We first assume that $ep(2)$ drops to 0.9, while all the other error propagation probabilities remain unchanged. This change brings the system reliability Rel from 0.4745 to 0.6078, with an increase of 28%. If we instead drop the value of a non-critical component, for example $ep(6)$, to the same value 0.9, while leaving unchanged the other ones, then Rel goes from 0.4745 to 0.4939, with an increase of only 4%.

With a similar logic, if we assume that $intf(8)$ decreases by the 20% of its value, that is from 0.1001 to 0.0801, then we obtain an improvement of Rel by 7% from 0.4745 to 0.5085. On the contrary, if we decrease $intf(5)$ (i.e. a non-critical component one) by the same percentage, from 0.01 to 0.008, then we obtain only a 0.17% improvement of Rel from 0.4745 to 0.4753.

In Figures 3 and 4 we report the derivatives of the system reliability with respect to the internal failure probabilities over their ranges, partitioned respectively as critical components (i.e., C_2, C_4, C_7 and C_8) and non-critical ones (i.e., C_1, C_3, C_5 and C_6) . Each curve has been obtained by varying the value of $intf(i)$ from 0 to 1 with a 0.1 step for all the components.

Similarly, in Figures 5 and 6 we report the derivatives of the system reliability with respect to the error propagation probabilities over their ranges.

In general, it is interesting to note that, for all the components, the function $\frac{\partial Rel}{\partial ep(i)}$ is monotonically decreasing whereas $\frac{\partial Rel}{\partial intf(i)}$ is monotonically increasing. This brings to find the highest absolute values of derivatives, respectively, near the value $ep(i) = 1$ for $\frac{\partial Rel}{\partial ep(i)}$ and near the value $intf(i) = 0$ for $\frac{\partial Rel}{\partial intf(i)}$.

$\frac{\partial Rel}{\partial intf(i)}$ for critical components in Figure 3 indicates that it is worth to work on the internal failure of a component when its probability falls close to 0, because large improvements can be induced on the system reliability in that subrange. On the contrary, it is not worth spending any effort to decrease the internal failure probability of components with high values of $intf(i)$ (i.e. close to 1), because no large gains can be obtained on the whole system reliability in that subrange. Figure 4 shows that the derivatives for non-critical components are almost always flat and very close to 0, thus changes in these components would never bring perceivable effects on the system reliability.

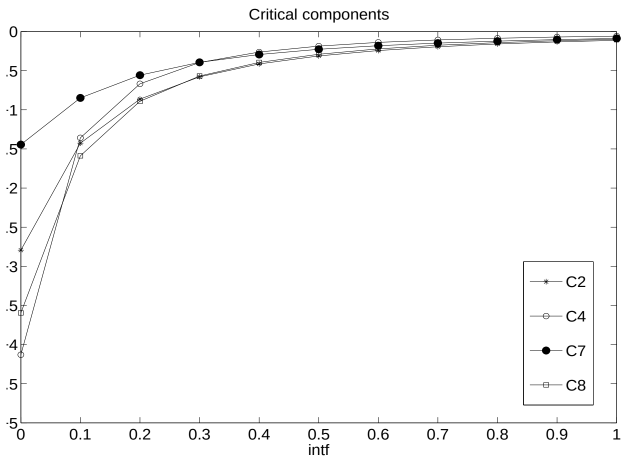

Figure 3: Derivatives of system reliability vs probability of failure: critical components.

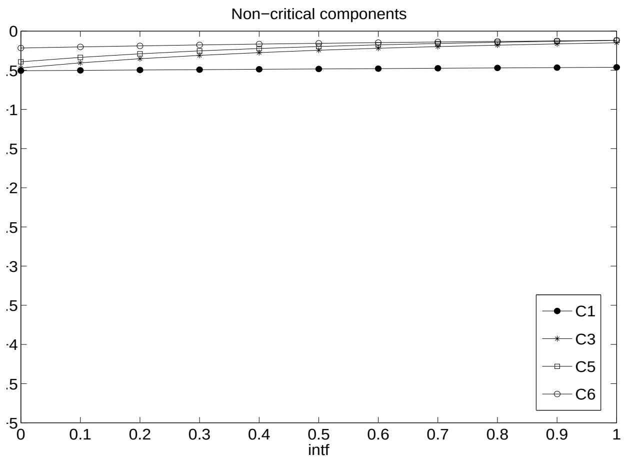

Figure 4: Derivatives of system reliability vs probability of failure: non-critical components.

$\frac{\partial Rel}{\partial ep(i)}$ for critical components in Figure 5 indicates that it is worth to work on the error propagation of a component when its probability falls close to 1, because large improvements can be induced on the system reliability in that subrange. On the contrary, it is not worth spending any effort to decrease the error propagation probability of components with low values of $ep(i)$ (i.e. close to 0), because no large gains can be obtained on the whole system reliability in that subrange. Figure 6 shows that the derivatives for non-critical components are almost always flat and very close to 0, thus changes in these components would never bring perceivable effects on the system reliability also in this case.

On the practical side, with modern testing techniques it is practically almost impossible to produce a software component with an internal failure probability larger than 0.001. Thus, it is very likely to find software components with values of $intf()$ in the range where the derivative is very high. This means that ever more effort has to be spent on software component reliability, as slight variations in their internal failure probability may heavily affect the whole system reliability. Likewise, it is very likely to find software

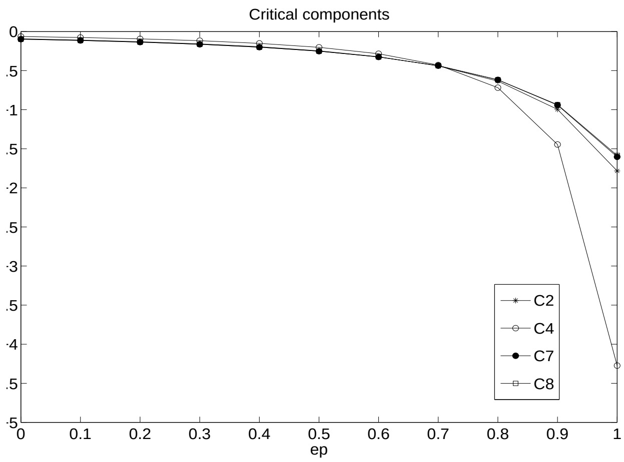

Figure 5: Derivatives of system reliability vs error propagation probability: critical components.

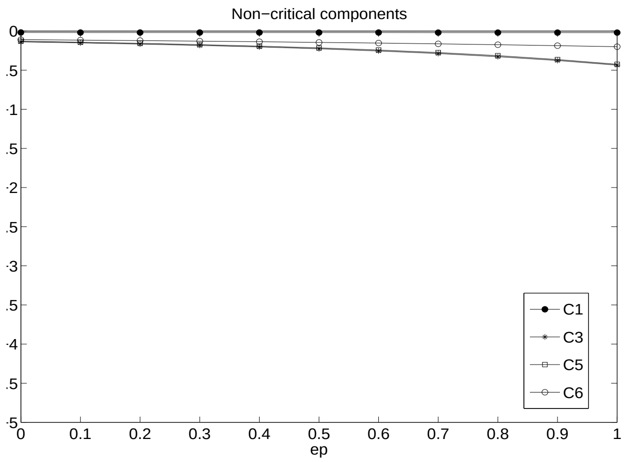

Figure 6: Derivatives of system reliability vs error propagation probability: non-critical components.

components with values of $ep()$ very close to 1 [12], that is in the range where the derivative is very high. Therefore, techniques that decrease the error propagation probability would similarly be suitable to sensibly affect the system reliability.

7 Conclusions

We have proposed an architectural approach to the reliability analysis of a modular software system that takes into account the impact of the error propagation characteristics of each component on the overall system reliability. We have shown that neglecting this impact may cause an imprecise prediction of the system reliability, with consequent wrong decisions about the most suitable system architecture. Moreover, our sensitivity analysis results can help in identifying the most critical system components, where the implementation and testing efforts should be focused, and where the placement of error detection and recovery mechanisms could be more effective.

Several open issues remain as future work, toward the goal of building a fully comprehensive architecture-based reliability model of a software system.

A first issue concerns the definition of a more complete architectural model, where, besides components, also connectors are taken into account. According to the software architecture paradigm, a connector is intended to embody all the aspects concerning the connection between offered and required services of components, so providing the modeling tool to explicitly represent all those aspects [3, 17]. The importance of explicitly modeling and taking into account connectors in the analysis of functional and extra-functional properties (like reliability) of a given architecture is widely recognized [2, 13, 18, 23]. Although we do not explicitly cope with connectors in our model, we believe that their error probabilities might sensibly affect, in some cases, the reliability of the whole system architecture.

A second issue concerns the adoption of a more comprehensive failure model. Given our focus on the analysis of the error propagation impact, we have only considered failures that generate erroneous output. However, other kinds of failures could affect the overall system reliability, like "stopping failures" that lead to a complete system stop [14]. In this respect, our model focuses on non-stopping failures. Moreover, we have not considered explicitly hardware failures, although in some cases they may heavily affect the reliability of a software system. We do not have considered all these issues in the model presented in this paper mainly because our intent has been to keep it as simple as possible in order to make our prime contribution (i.e. error propagation modeling) clearly emerging. We point out that in a previous work [9] we have proposed an architectural model that embeds connectors and stopping software/hardware failures, without keeping into account error propagation. We are working towards the merging of the model presented in [9] an the one presented in this paper.

Finally, an important issue concerns the trade-off between model tractability and model refinement. In this respect, it is worth investigating the most suitable granularity level in component reliability modeling: in particular, modeling the individual offered service reliability versus averaging their reliability into component-level reliability parameters, and modeling input-dependent behavior versus averaging over all the possible input-dependent behaviors.

References

[1] W. Abdelmoez, D.M. Nassar, M. Shereshevsky, N. Gradetsky, R. Gunnalan, H.H. Ammar, B. Yu, A. Mili, Error Propagation in Software Architectures, Proc. of the 10th International Symposium on Software Metrics, Chicago, USA, September 2004.

[2] R. Allen, D. Garlan, A formal basis for architectural connection, ACM Trans. on Software Engineering and Methodology, vol. 6, no.3, pp.213-249, July 1997.

[3] L. Bass, P. Clements, R. Kazman, Software Architectures in Practice, Addison-Wesley, New York, NY, 1998.

[4] R. C. Cheung, A user-oriented software reliability model, IEEE Trans. on Software Engineering, 6(2):118–125, March 1980.

[5] E. Cinlar, Introduction to Stochastic Processes, Prentice-Hall, 1975.

[6] J. Dolbec, T. Shepard, A component based software reliability model, Proc. of the 1995 Conference of the Centre for Advanced Studies on Collaborative Research (CASCON), Toronto, Ontario, Canada, 1995.

[7] S. Gokhale, W.E. Wong, J.R. Horgan, K. Trivedi, An analytical approach to architecture-based software performance and reliability prediction, Performance Evaluation, n.58 (2004), pp. 391-412.

[8] K. Goseva-Popstojanova, A.P. Mathur, K.S. Trivedi, Architecture-based approach to reliability assessment of software systems, Performance Evaluation, no. 45 (2001), pp. 179-204.

[9] V. Grassi, Architecture-based Reliability Prediction for Service-oriented Computing, Architecting Dependable Systems III (R. de Lemos, A. Romanovsky, C. Gacek Eds.), LNCS 3549, Springer-Verlag, 2005, pp. 279-299.

[10] V. Grassi, V. Cortellessa, Embedding error propagation in reliability modeling of component-based software systems, International Conference on Quality of Software Architectures, NetObjectDays'05, Erfurt, Germany, Sept. 20-21, 2005.

[11] D. Hamlet, D. Mason, D. Woit, Theory of software reliability based on components, Proc. of 23rd Int. Conference on Software Engineering (ICSE 2001),Toronto, Canada, May 2001.

[12] M. Hiller, A. Jhumka, N. Suri, EPIC: Profiling the Propagation and Effect of Data Errors in Software, IEEE Trans. on Computers, vol. 53, no.5, pp.512-530, May 2004.

[13] Inverardi, P. and Scriboni, S., Connectors Synthesis for Deadlock-Free Component-Based Architectures, Proc. of Automated Software Engineering Conference, 2001.

[14] J.C. Laprie (ed.), Dependability: Basic Concepts and Terminology, Springer-Verlag, 1992.

[15] S. Krishnamurthy, A.P. Mathur, On the estimation of reliability of a software system using reliabilities of its components, in: Proceedings of the Eighth International Symposium on Software Reliability Engineering (ISSRE'97), 1997, pp. 146-155.

[16] D. Mason, Probabilistic analysis for component reliability composition, 5th ICSE Workshop on Component-based Software Engineering, May 2002.

[17] N.R. Mehta, N. Medvidovic, S. Phadke, Toward a taxonomy of software connectors, Proc. 22nd Int. Conference on Software Engineering, May 2000.

[18] R.H. Reussner, H.W. Schmidt, I.H. Poernomo, Reliability prediction for component-based software architectures, Journal of Systems and Software, no. 66, 2003, pp. 241-252.

[19] R. Roshandel, N. Medvidovic, Toward architecture-based reliability prediction, Proc. ICSE 2004 Workshop on Architecting Dependable Systems (WADS 2004), (R. de Lemos, C. Gacek, A. Romanowsy eds.), Edinburgh, Scotland, UK, May 2004, pp. 2-6.

[20] N.D. Singpurwalla, S.P. Wilson, Statistical Methods in Software Engineering, Springer Series in Statistics, 1999.

[21] J.Voas, PIE: A Dynamic Failure-Based Technique, IEEE Trans. on Software Engineering, Vol. 18, No. 8, pp. 717-727, 1992.

[22] S. Yacoub, B. Cukic, H. Ammar, Scenario-based reliability analysis of component-based software, Proc. of the 10th International Symposium on Software Reliability Engineering, 1999, pp. 22-31.

[23] W.-L. Wang, Y. Wu, M.-H. Chen, An architecture-based software reliability model, Proc. IEEE Pacific Rim Int. Symposium on Dependable Computing, Hong Kong, China, Dec. 1999.

Appendix

In this appendix we present the theorem proofs that have not been reported in the paper text. Some of them (i.e. Theorems 1, 2 ad 3) have already been presented in [10].

Proof of Theorem 1.

Let us define the following events:

- $\texttt{transf(i,j)}$: component i *directly* transfers the control to component j (i.e. in one only step);
- $\texttt{reach}^{(k)}\texttt{(i,j)}$: component j gets control exactly at the k-th step, given that the execution started at component i;
- $\texttt{errout}^{(k)}\texttt{(i)}$: component i produces an erroneous output, given that it gets control at the k-th step;
- $\texttt{intfail}^{(k)}\texttt{(i)}$: an internal failure occurs in component i, given that it gets control at the k-th step;

- $\texttt{errprop}^{(k)}\texttt{(i)}$: the component i propagates an erroneous input to its output, given that it gets control at the k-th step.

We have by definition:

$$Pr\{\texttt{transf(i,j)}\} = p(i,j),$$
$$Pr\{\texttt{reach}^{(k)}\texttt{(i,j)}\} = p^{(k)}(i,j). \tag{22}$$

Moreover, by the assumption of independence of the internal failure and error propagation probabilities from the past history, we have:

$$(\forall k) Pr\{\texttt{intfail}^{(k)}\texttt{(j)}\} = if(j),$$
$$(\forall k) Pr\{\texttt{errprop}^{(k)}\texttt{(j)}\} = ep(j). \tag{23}$$

Let us consider the joint event $\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)}\}$. We have:

$$\begin{aligned} err^{(k)}(i,j) &= \\ &= Pr\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)}\} \\ &= Pr\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)} \wedge \texttt{intfail}^{(k)}\texttt{(j)}\} \\ &+ Pr\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)} \wedge \overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \end{aligned} \tag{24}$$

The probability of the joint event $\{\texttt{errout}^{(k)}\texttt{(j)} \wedge \texttt{intfail}^{(k)}\texttt{(i,j)}\}$ can be expressed as

$$\begin{aligned} & Pr\{\texttt{errout}^{(k)}\texttt{(j)} \wedge \texttt{intfail}^{(k)}\texttt{(i,j)}\} \\ & = Pr\{\texttt{errout}^{(k)}\texttt{(j)} \mid \texttt{intfail}^{(k)}\texttt{(i,j)}\} \cdot Pr\{\texttt{intfail}^{(k)}\texttt{(i,j)}\} \\ & = Pr\{\texttt{intfail}^{(k)}\texttt{(i,j)}\} \end{aligned} \tag{25}$$

On the other end, it is easy to realize that the probability of the joint event $\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)} \wedge \overline{\texttt{intfail}^{(k)}\texttt{(j)}}\}$ can be expressed as:

$$\begin{aligned} & Pr\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)} \wedge \overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \\ & = Pr\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)} \mid \overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \cdot Pr\{\overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \end{aligned}$$

By observing that the event $\texttt{errout}^{(k)}\texttt{(j)}$, given the absence of internal failures in component j (i.e. $\overline{\texttt{intfail}^{(k)}\texttt{(j)}}$), can only be originated by error propagation from previous steps, we can modify the above formula as follows:

$$\begin{aligned}
&Pr\{\texttt{reach}^{(k)}\texttt{(i,j)} \wedge \texttt{errout}^{(k)}\texttt{(j)} \mid \overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \cdot Pr\{\overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \\
&= \sum_{h=0}^{C} Pr\{\texttt{reach}^{(k-1)}\texttt{(i,h)} \wedge \texttt{errout}^{(k-1)}\texttt{(h)} \wedge \texttt{transf(h,j)} \wedge \texttt{errprop}^{(k)}\texttt{(j)}\} \cdot \\
&\cdot Pr\{\overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \\
&= \sum_{h=0}^{C} err^{(k-1)}(i,h) \cdot Pr\{\texttt{transf(h,j)} \wedge \texttt{errprop}^{(k)}\texttt{(j)}\} \cdot Pr\{\overline{\texttt{intfail}^{(k)}\texttt{(j)}}\} \qquad (26)
\end{aligned}$$

Using equations (25) and (26), we can rewrite equation (24) as follows:

$$\begin{aligned}
err^{(k)}(i,j) &= \\
&= Pr\{\texttt{reach}^{(k)}\texttt{(i,j)}\} \cdot Pr\{\texttt{intfail}^{(k)}\texttt{(j)}\} \\
&+ \sum_{h=0}^{C} err^{(k-1)}(i,h) \cdot Pr\{\texttt{transf(h,j)} \wedge \texttt{errprop}^{(k)}\texttt{(j)}\} \cdot Pr\{\overline{\texttt{intfail}^{(k)}\texttt{(j)}}\}
\end{aligned}$$

Finally, the theorem is proven by substituting in latter expression the event probabilities given by (22) and (23). **qed**

Proof of Theorem 2.

The proof is by induction on k.

Given the assumption in the hypothesis of Theorem 1, for $k = 0$ we have from equation (5):

$E^{(0)} = Q^0 \cdot F = F$,

which is the value of the right-hand side of (6), namely

$\sum_{i=0}^{0} \mathbf{Q}^{0-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i$.

Let the theorem hold for $k-1$. From (5) and the induction assumption we get:

$$\begin{aligned}
\mathbf{E}^{(k)} &= \\
&= \mathbf{Q}^k \cdot \mathbf{F} + \mathbf{E}^{(k-1)} \cdot \mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}) \\
&= \mathbf{Q}^k \cdot \mathbf{F} + \sum_{i=0}^{k-1} \mathbf{Q}^{k-1-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i \cdot \mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}) \\
&= \mathbf{Q}^k \cdot \mathbf{F} + \sum_{i=0}^{k-1} \mathbf{Q}^{k-1-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^{i+1} \\
&= \mathbf{Q}^k \cdot \mathbf{F} + \sum_{i=1}^{k} \mathbf{Q}^{k-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i
\end{aligned}$$

$$= \sum_{i=0}^{k} \mathbf{Q}^{k-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i.$$

qed

Proof of Theorem 3.

$$\begin{aligned}
\mathbf{e} &= \\
&= \sum_{k=0}^{\infty} \sum_{i=0}^{k} \mathbf{Q}^{k-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i \cdot \mathbf{c} \\
&= \sum_{i=0}^{\infty} \sum_{k=i}^{\infty} \mathbf{Q}^{k-i} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i \cdot \mathbf{c} \\
&= \sum_{i=0}^{\infty} [(\sum_{k=i}^{\infty} \mathbf{Q}^{k-i}) \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i] \cdot \mathbf{c} = \\
&= \sum_{i=0}^{\infty} [(\sum_{k=0}^{\infty} \mathbf{Q}^{k}) \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i] \cdot \mathbf{c} \\
&= \sum_{i=0}^{\infty} [(\mathbf{I} - \mathbf{Q})^{-1} \cdot \mathbf{F} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i] \cdot \mathbf{c} \\
&= (\mathbf{I} - \mathbf{Q})^{-1} \cdot \mathbf{F} \cdot \sum_{i=0}^{\infty} (\mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^i \cdot \mathbf{c} \\
&= (\mathbf{I} - \mathbf{Q})^{-1} \cdot \mathbf{F} \cdot (\mathbf{I} - \mathbf{Q} \cdot \mathbf{R} \cdot (\mathbf{I} - \mathbf{F}))^{-1} \cdot \mathbf{c}.
\end{aligned}$$

qed

Proof of Theorem 6.

The proof is by induction. For $k = 0$ the theorem holds true, since the last summation in the right hand side of (20) is empty, and by definition $\tilde{\mathbf{E}}^{(0)} = \mathbf{A}^{(0)}$. Let the theorem holds true for $k - 1$; we have:

$$\begin{aligned}
\tilde{\mathbf{E}}^{(k)} &= \\
&= \tilde{\mathbf{E}}^{(k-1)} \mathbf{QR}(\mathbf{I} - \mathbf{F}) + \mathbf{A}^{(k)} - \mathbf{D}^{(k-1)}(\mathbf{R}) \\
&= (\textit{by induction hypothesis}) \\
&\quad (\sum_{i=0}^{k-1} \mathbf{A}^{(k-i-1)} (\mathbf{QR}(\mathbf{I} - \mathbf{F}))^i - \sum_{i=0}^{k-2} \mathbf{D}^{(k-i-2)} \mathbf{R} (\mathbf{QR}(\mathbf{I} - \mathbf{F}))^i) \mathbf{QR}(\mathbf{I} - \mathbf{F}) + \mathbf{A}^{(k)} - \mathbf{D}^{(k-1)} \mathbf{R} \\
&= \sum_{i=0}^{k-1} \mathbf{A}^{(k-i-1)} (\mathbf{QR}(\mathbf{I} - \mathbf{F}))^{i+1} - \sum_{i=0}^{k-2} \mathbf{D}^{(k-i-2)} \mathbf{R} (\mathbf{QR}(\mathbf{I} - \mathbf{F}))^{i+1} + \mathbf{A}^{(k)} - \mathbf{D}^{(k-1)} \mathbf{R} \\
&= \sum_{i=0}^{k} \mathbf{A}^{(k-i)} (\mathbf{QR}(\mathbf{I} - \mathbf{F}))^i - \sum_{i=0}^{k-1} \mathbf{D}^{(k-i-1)} \mathbf{R} (\mathbf{QR}(\mathbf{I} - \mathbf{F}))^i
\end{aligned}$$

qed

Proof of Theorem 7.

Let us define $\tilde{\mathbf{E}} = \sum_{k=0}^{\infty} \tilde{\mathbf{E}}^{(k)}$. We have from Theorem 6:

$$
\begin{aligned}
\tilde{\mathbf{E}} &= \sum_{k=0}^{\infty} \tilde{\mathbf{E}}^{(k)} = \\
&= \sum_{k=0}^{\infty} (\sum_{i=0}^{k} \mathbf{A}^{(k-i)} (\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i - \sum_{i=0}^{k-1} \mathbf{D}^{(k-i-1)} \mathbf{R} (\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i) \\
&= \sum_{i=0}^{\infty} \sum_{k=i}^{\infty} \mathbf{A}^{(k-i)} (\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i - \sum_{i=0}^{\infty} \sum_{k=i+1}^{\infty} \mathbf{D}^{(k-i-1)} \mathbf{R} (\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i \\
&= \sum_{i=0}^{\infty} (\sum_{k=0}^{\infty} \mathbf{A}^{(k)}) (\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i - \sum_{i=0}^{\infty} (\sum_{k=0}^{\infty} \mathbf{D}^{(k)}) \mathbf{R} (\mathbf{QR}(\mathbf{I}-\mathbf{F}))^i \\
&= (\sum_{k=0}^{\infty} \mathbf{A}^{(k)}) (\mathbf{I} - \mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1} - (\sum_{k=0}^{\infty} \mathbf{D}^{(k)}) \mathbf{R} (\mathbf{I} - \mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1} \\
&= (\mathbf{A} - \mathbf{DR})(\mathbf{I} - \mathbf{QR}(\mathbf{I}-\mathbf{F}))^{-1}
\end{aligned}
$$

The derivation of $\mathbf{D}$ has been given in Theorem 5, whereas the definition of $\mathbf{A}$ given in this theorem follows from the following considerations: (i) $\mathbf{A}^{(k)}$ is a diagonal matrix built from the entries of row 0 of the matrix $\mathbf{Q}^{(k)}$, and (ii) $\sum_{k=0}^{\infty} \mathbf{Q}^{(k)} = (\mathbf{I} - \mathbf{Q})^{-1}$. Finally, the expression for $\tilde{\mathbf{e}}$ given in this theorem follows from (19). **qed**