Architecture-Based Software Reliability Prediction Approach for Component Based Software

Awad Ali and Dayang N.A. Jawawi
Department of Software Engineering, Faculty of Computing
Universiti Teknologi Malaysia
UTM, 81310 Skudai, Johor, Malaysia
awad.uoftkassala@gmail.com, dayang@utm.my

Abstract—Computer software is playing central role in our daily life. Most of industries in various disciplines depend on computer software for their basic functioning. The industries in these disciplines always request high-quality software. The quality of software depends on certain attributes which include reliability, dependability, usability, flexibility, performance, safety, interoperability and security. The reliability attribute is prioritized as the most important quality attributes. Currently, the challenge being faced by reliability is how it can be conducted at the design’s time before the system implementation, while the sources of information for the reliability are limited. Two major problems are, non-availability of technical data about the failure, and the operational profile is not developed yet. This paper defines the whole elements that are needed by the reliability prediction approach of component based software. The research focuses in proposing an initial work of probabilistic dependency graph models as part of the approach. The graph and the parameters are employed as part of the prediction approach. The graph represents and links the system components and their internal states through a set of parameters.

Keywords—Reliability prediction, software architecture, component software.

I. INTRODUCTION

Nowadays, Computer software is playing vital roles in many disciplines. It’s embedded in cars, hand phone, automobile, and aircraft. In addition, the software is increasingly used to support critical business applications and industrial processes. Most of these disciplines depend on the software for their basic functioning. Failure of software can lead to critical, event fatal, consequences in safety-critical applications as well as in business applications. In order to meet a customer’s expectations and needs, the software must have high reliability. An increasing demands on software functionalities is leading to issues which include, scalability, degree of concurrency of current software system the customer’s satisfaction as serious challenge; thus, the reliability engineering should live up to these today’s complex software systems and their specific challenges.

The reliability approach is formalized to explain the failure behavior of the system. Software reliability is defined as the probability that the software system will perform a required function correctly (failure free) in a stated environment for a specified period of time. Due to the heterogeneity of execution environment and the development methodology of the current software systems, a failure broadly can means, that the software system is unable to deliver the expected service and not capable to resume its service as it was not interrupted. Several kinds of failures are faced during service execution such as faults in the implementation of the software components, hardware failure and network failure. Hardware failure is due to unreliable hardware resource, and network failures is just because of message loss or problem of inter component communication [1]. Predicting software reliability at an early design level enables the software’s designer to identify weak design spots, which would be cost effective to improve than fixing consequent errors at later implementation phases. Therefore, reliability approach must be able to work at design and development stage, and particularly during architectural design. A main challenge for predicting the reliability at the design time is the lack of information about the operational profile and failure information of the components. For example during software architecture design, Software’s designers may design new components or use existing components. In case of existing components, the reliability is estimated from their previous operational use of the components in other software systems. But in case of new components, no such information will be available about the components except information of design models software architecture and design represents critical decisions that govern software functionality [2]. In fact, there are three aspects have been described as motivations for software architectural design. First, it offers an artifact that initiates support for arguments by the stakeholders at very early stage of the design time. Second, it allows for early judgment of quality attributes. Finally, the decisions captured in the software architecture can be transferred to other systems [3]. Software architecture is defined as the structure or structures of the system, which comprise software elements such as components and their properties, the externally visible properties of those components and their connectors. Furthermore it describes the relationships among all these elements [4]. The system view provided by an architectural description is valuable throughout the life of a software system as an abstract reference point for evaluation and change [5]. The reliability approach that builds upon software architecture, allow understanding of the influence of early architectural decisions on the system’s dependability. When reliability approach works early as evaluation mechanism prior to the implementation and testing of software, it can reveal whether the architecture meets the behavioral and reliability requirements or not. The early
evaluation will enable the reliability issues to be solved simply at the design time. So the reliability approach should sufficiently leverage the available architecture-level knowledge about the system, and does not only rely on implementation level data that leads to the system’s failure behavior.

II. RESEARCH MOTIVATION AND CHALLENGES

According a number of surveys papers [3, 5-9] there are some challenges reported in the literature related to current architecture base approaches for example:

- While architecting, reliability of software architecture is governed by the reliability of individual component, components interactions, and the execution environment, information about the reliability of the components might be not exist yet. Thus, the reliability approach should not assume that all components reliabilities are known. Consequently, the approach should provide mechanisms for obtaining such information. Furthermore the individual component internal structure and behavior should consider as basic building block in this mechanism toward estimating components reliabilities.

- With increasing of software functionalities and complexity, and the number of components and others architecture’s elements, the reliability approach should demonstrate high scalability to manage the concurrency execution and be able to deal with a big number of system’s components. Moreover, the consideration of individual component internal structure will increase the prediction complexity, for example, if the architecture comprise \( i \) components and each component have \( j \) states the prediction formula will deal with \( i^j \) states that, if the reliability prediction traced hierarchically and \( i*j \) in case of sequentially [5, 10].

- The uncertainty in system behavior at the design time that leads to uncertainty in reliability parameters which is “not getting much attention” [11]. For example, the software system often has interactions with external systems or services, which are not presented with an accurate specification. Another aspects, the actual system behavior at execution time is inferred by system usage, which may be hard to predict at design time [1, 11]. The reliability approach should consider such situations.

III. PROBLEM STATEMENT

The architecture is an important part that should be considered to predict the reliability of the software system. In the context of architecture, various approaches and techniques have been developed. A number of the current approaches assume that obtaining operational profile at the design time is very difficult, since they used the profile in terms of usage profile to estimate the transition probability between components and the component’s internal states. However the usage profile does not reveal the actual situation precisely and that lead to unrealistic prediction. Limited studies use the profile in term of operational profile, while they have some limitations. These limited studies depended on hidden markov model as mathematical model for profile modeling and used hierarchical scenario method in the system scenario. However the use of hierarchical hidden markov model may it more consistent with the scenario’s method, hence it can provide more accurate transition values than hidden markov model. Consequently operational profile that provides accurate transition values can enable realistic synthesized scenarios towered accurate prediction. Moreover there is need for prediction technique that can expose the system and its internal parts in connected skeleton. The connected skeleton can allow the calculation of the reliability of system’s scenario to be tractable, if we manage the scenarios well by truncating them to subscenarios. This situation is leading to the primary research question for this study:

“How to develop an accurate prediction approach for component based systems’ reliability at design time?”

This main research question is broken down and can be complimented by the following sub-questions:

- How the architecture of the software can be exposed in structural and behavioral context in the reliability prediction?
- How the architecture analysis results can be mapped to reliability prediction?
- How the prediction will conduct depending on the architecture analysis results and the architecture artifact?
- How the operational profile of the system can be estimated at the design time?
- How the system’s scenarios can be synthesized with a consideration of component internal states of each component in each scenario?
- How to evaluate the accuracy of the proposed approach?

This research paper answers only the first three questions.

IV. RESEARCH AIMS AND OBJECTIVES

The aim of this research is to develop an accurate prediction approach for the reliability of software system based on the architectural design. The objectives of this research work can be organized as follows:

- To propose dependency graph that can represent the system structures, which include the individual component’s internal states, connectors, and system components and combine architecture analysis results in the representation.
- To propose reliability prediction algorithm that can incorporate component reliabilities in the system reliability using the graph parameters and data provided by representative operational profile and system scenarios.

SOFTWARE TESTING CONFERENCE (SoftecAsia2013) (Sunway Resort Hotel &Sa, Kuala Lumpur 2 September 2013)
ISBN 978-967-012-075-1
To develop a representative operational profile which estimates transition probabilities between both component’s internal states and the system components?

To develop a method which can be used to describe system scenarios and the internal component scenarios?

To demonstrate and measure the accuracy of the proposed approach by applying the proposed approach in the selected applications as case studies.

In this paper an initial work is proposed for objectives one and two. In fact, for objective two, we are focusing in the related parameters that will be used in the prediction algorithm.

V. RELATED WORKS

During the last ten years, researchers have proposed many approaches for predicting the reliability depending on the architectural design; these approaches are addressing different problems and challenges. However, Individual component reliability is an important part that should be considered to predict the reliability of the software system. Most of the current approaches estimate the reliability of the system as a function of the reliabilities of its components, without going into sufficient details regarding the internal behaviors of the individual component toward predicting the reliability. To check out the above mentioned situation and the ability to tackle the current challenges, we have developed the following five questions; that are used as evaluation mechanism to analyze the strengths and weaknesses of the current architecture based reliability prediction approaches.

Q1: Is the system reliability predicted depending on the reliabilities of individual components?

Q2: Are the interactions between components considered?

Q3: Does the approach depend on operational profile?

Q4: Can the approach be used in case of new brand components?

Q5: Does the approach propose solutions for the complex scenarios?

The results of our evaluation depending on these questions are shown in Table 1.

From Table 1, we can see the evaluation results of the approaches and the research gaps. Through Q1 and Q2, it is found that all the approaches estimates the reliability as a function of reliability of individual component and consider the interactions between the components. This result describes the importance of the reliability of the individual component and the interactions between components in the system’s reliability. Q3 exposes that most of the approaches depend on the operational profile except 4 approaches. These 4 approaches use the usage profile for estimating the transition probabilities between components. Usage profile includes a cross-product of all key input variables with their possible values and occurrence probabilities. These values are generated by the probability distribution. Later after the implementation stage, these occurrence probabilities will map to operations, which capture a specific behavior on the implementation level [12, 13]. However the usage profile does not reveal the actual situation precisely, it provides approximate values. For this reason, the scenarios build upon usage profile do not describe the component’s dynamic behavior precisely [5, 6]. As a result from Q3, it is clear that the operational profile is essential for the reliability prediction. Furthermore, the approaches assume that there are operational data for the individual component. Sometimes such information is not available in case of a brand new component. Software engineers may design new components or integrate existing components. In order to measure the reliability of the system based on the individual component reliabilities’ values, there are many issues which must be considered. These issues are availability of component code, testing data, and whether the component is new or a reused. Previous situations show that most of the approaches designed to deal with the components already have operational data and due to this logical reason they do not address operational profile as shown in Q4. From Q4 it is clear that, there are only 3 approaches considering a brand new component in their prediction. Q5 elaborates that limited approaches propose solutions to deal with complex scenarios. Furthermore, only one of these approaches is addressing operational profile’s problem.

TABLE 1. THE EVALUATION RESULTS OF ARCHITECTURE BASED RELIABILITY APPROACHES

<table>
<thead>
<tr>
<th>The Approach</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Singh et al. [14]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Godbole &amp; S. Trivedi. [15]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Cortellissa et al. [16]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Reussner et al. [17]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>Yacoub et al. [18]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>B. Cukic [19]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Rodrigues et al. [20]</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>√</td>
</tr>
<tr>
<td>Rosshandel et al. [21]</td>
<td>√</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Zhang [22]</td>
<td>√</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>L. Cheung et al. [23]</td>
<td>√</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Dong et al. [24]</td>
<td>√</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>√</td>
</tr>
<tr>
<td>Goswami and Acharya [25]</td>
<td>√</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hsu and Huang [26]</td>
<td>√</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>M. Palviainen et al. [27]</td>
<td>√</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>√</td>
</tr>
<tr>
<td>Brosch et al. [1]</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Tyagi and Sharma [28]</td>
<td>√</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Kim et al. [4]</td>
<td>✓</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>

✓ Requirement satisfied  
× Requirement unsatisfied

The previous discussion states that most of the current approaches assume that all components’ reliabilities are known; so there is no approach which can be used when we consider a brand new component accept the approaches [21, 23]. The operational profile is main problem when we consider a brand new component at the design time, because the component has not yet been built. The representative operational profile may help to synthesize realistic or accurate
scenarios and solve uncertainties about the exact functionality of the system and its components to make the reliability approach truly meaningful, useful and usable. Furthermore, the excepted approaches [21, 23] are facing scalability problem when considering the complex and large systems.

VI. RESEARCH METHODOLOGY

This research work contains three stages: reliability prediction techniques stage which contains construction of dependency graph and prediction algorithm, operational profile modeling stage and system’s scenarios synthesizing stage. Fig. 1 shows flowchart which reflects the stages and activities that will be conducted sequentially as part of the research work. In Fig. 1 stages three and four are not in the scope of this paper, however the stages are shown here to define the whole part that will support the prediction technique. This paper focuses only on the Stage 1 and Stage 2. In Stage 1 we explore the challenges faced during reliability prediction at the design time, the problems related to mapping system’s architecture in reliability prediction, and architecture based reliability prediction approaches and their limitations. Stage 2 of Fig. 1 describes the architecture parameters that are required for the prediction. The parameters must reveal the behavior of component and the whole system. These parameters are used in the prediction algorithm. Furthermore, the stage 2 describes how the architecture analysis result will map to the reliability prediction and how the proposed reliability prediction technique can join the components and internal states of the component in the prediction. Furthermore, in this stage we will develop an algorithm to calculate the reliabilities of components and the whole system (this activity is also outside the scope of this paper). In addition the chart describes the evaluation activities that will be conducted at the end of each stage’s work.

VII. INITIAL WORK OF THE RELIABILITY APPROACH

Graphs and causal semantics which are known as probabilistic dependency graphs models, are a very useful means to interpret and describe complex structures and systems and to model concepts and ideas in a direct and intuitive way. This section works introduce dependency graph as part of our prediction approach. The graph will be used in order to represent static and dynamic aspects of the software architecture of the system. Furthermore, we will define the graph’s notation and parameters.

A. Inner and Outer Probabilistic Dependency Graph (IOPDG)

An Inner and Outer Probabilistic Dependency Graph is an annotated graph that in one hand exposes the probabilistic relationships among system components, and on the other hand it reveals the internal component’s behavior which govern the component reliability. The use of probabilistic graph is classical method in software engineering applications [29]. Baah [29] proposed probabilistic graphical model works with algorithms to analyze program behaviors. This model is used for program’s fault comprehension and localization. In the architecture based reliability estimation and analysis, there is a limited usage for the probabilistic graph. Yacoub et al. [18] proposed probabilistic dependency graph named component dependency graph (CDG) with scenario based algorithm as technique to analyze the reliability of component based software. The IOPDG which showed in Fig 2 and Fig 4 differs from CDG due to difference in individual component’s internal structure, which is considered in IOPDG. The internal behavior may cause failure during component operations and reduce the reliability of component and hence, effect on the overall software reliability [30, 31]. IOPDG connect two spaces, the first space contains the component’s internal states which will expose the internal static and dynamic behaviors and the internal interactions. The second space represents the system components and the interaction between them. The connection among these two spaces can support the idea of meaningful subsystems which is an active subject in the field of search-based software.

![Fig. 2. One Component with Internal states](image-url)

![Fig. 3. IOPDG’s keys](image-url)
Fig. 2 shows how each component can be represented depending on the internal states and the failure states. The internal states are named s1, s2, and so, the failure state (black nodes) are named FS1 and FS2. Fig. 4 shows how the component connected together toward constructing IOPDG. Fig. 3 defines keys to understand interactions’ types among the internal states of IOPDG.

B. IOPDG Basic Notation and Definitions

IOPDG is a graph contains two structure graphs G1 and G2.

\[ \text{G1} = (N_1, E_1) \]

is subgraph of a graph \[ \text{G2} = (N_2, E_2), \]

the map from G1 to G2 is defined by \( \eta: \text{G1} \rightarrow \text{G2} \) which satisfies: \( \forall n_1 \in N_1, \ n_1 \) maps to one and only one \( n_2 \in N_2 \). denoted by \( \eta(n_1) = n_2 \) where:

- \( N_1 \) is a finite set of nodes represents the component’s internal states
- \( E_1 \) is a finite set of edges represent the interactions between the component’s internal states
- \( N_2 \) is a finite set of nodes represents the system’s components
- \( E_2 \) is a finite set of edges represent the interactions between the components

Definition 1: G1’s elements

- \( N_1 \) is defined by the tuple \( \text{CS}_{ij}, \text{FS}_i, \text{RCS}_{ij}, \text{RC}_{ij}, \text{PEFS}_{ij}, \text{head}, \text{rear} \) where:
  - \( \text{CS}_{ij} \): is a unique identification identifying states (state \( j \) in component \( i \)).
  - \( \text{FS}_i \): is a unique identification identifying failure states.
  - \( \text{RCS}_{ij} \): is a reliability of state.
  - \( \text{RC}_{ij} \): is a reliability of state corresponding to the failure state \( j \).
  - \( \text{PEFS}_{ij} \): is a probability of recovery from the failure state \( j \).
  - \( \text{RCS}_{ij} \): is a reliability of a state (it is a probability value indicates that the component will pass the current state correctly (fault free) corresponding to all failure states.
- \text{head} is a pointer points to the start state, \text{rear} is a pointer points to the termination state.

- \( E_1 \) is defined by the tuple \( \text{ST}_{ik}, \text{TY}_{jk}, \text{PT}_{ij} \) where:
  - \( \text{ST}_{ik} \): is a unique identification identifying a state transition (the transition from state \( i \) to state \( k \)).
  - \( \text{TY}_{jk} \): (\( \text{NS} \rightarrow \text{FS} \) | \( \text{NS} \rightarrow \text{NS} \) | \( \text{FS} \rightarrow \text{NS} \) ) it is nominal value used to specify the type of transitions between states where \( \text{NS} \rightarrow \text{FS} \) Transition from normal state to failure state.

Fig. 4. Inner and Outer Probabilistic Dependency Graph (IOPDG)

VIII. CONCLUSION AND FUTURE WORK

The aim of this research work is to introduce an accurate prediction approach for the reliability of software system based on the architectural design. In this paper, we have proposed Probabilistic Dependency Graph as part of our objectives. The graph exposed how the probabilistic relationships among system components and the internal component’s states can be represented. Our ongoing work in this research is to develop an algorithm for predicting the reliability of the system based on the proposed graph parameters. Beside the graph, the algorithm will utilize operational profile data and scenario method to predict the reliability. As we mentioned in section IV the development of operational profile and scenarios method are part of our objectives in this research work which will be completed at later stages. However in order to evaluate the prediction algorithm before reaching to the second and the third step, we will use case study provided with operational profile data and scenarios which is used in the work [23].

ACKNOWLEDGEMENTS

We are very grateful to Ministry of Science, Technology and Innovation Malaysia (MOSTI) Vote No.4S064, and Universiti Teknologi Malaysia, for their financial support of this research. We would also like to thank Embedded & Real-Time Software Engineering Laboratory (EReTSEL) members for their feedback and continuous support.
REFERENCES


