Abstract

Recent evidences indicate that most faults in software systems are found in only a few of a system’s components [1]. The early identification of these components allows an organization to focus on defect detection activities on high risk components, for example by optimally allocating testing resources [2], or redesigning components that are likely to cause field failures. This paper presents a prototype tool called Architecture-level Risk Assessment Tool (ARAT) based on the risk assessment methodology presented in [3]. The ARAT provides risk assessment based on measures obtained from Unified Modeling Language (UML) artifacts [4]. This tool can be used in the design phase of the software development process. It estimates dynamic metrics proposed in [5] and automatically analyzes the quality of the architecture to produce architectural-level software risk assessment [3].

1. Introduction

There is an increasing need for a tool that can be used to track the quality of software products during the software design phase. Some of the proposed tools get static metrics from source code [6], but source code metrics are affected by the programming language and programming style. When calculating the metrics from architectural descriptions such as UML, we achieve independency of languages and human factors [7]. Other metrics tools [8] only produce static metrics to describe the model with a limited capability to accurately represent the dynamic behavior of the architecture. We developed a tool for risk assessment at the architectural level, ARAT, that produces the dynamic metrics from UML diagrams. Using the methodology proposed in [3], the tool estimates risk factor of a software system in hieratical fashion. This paper is organized as follow. Section 2 presents an overview of the tool structure. Section 3 provides some examples of the outputs from the tool. Finally, Section 4 concludes the paper.

2. Structure of the tool

The UML use case model of the tool is shown in Figure 1. The tool consists of six main use cases as follows.

**Estimate dynamic metrics.** Based on cyclomatic complexity [10], the ARAT computes the dynamic complexity [5] of the UML state charts as a dynamic complexity metric. It also computes dynamic coupling between components based on the message exchange between components in the sequence diagram [5].

**Collect model information.** Using a commercial tool Rose Real Time [9] as a front end for our tool we transform the visual UML model to a textual format data. We use Rose Script to transform UML diagrams to textual data.

**Estimate component / connector risk factor.** Based on the definition of risk [11], we calculate risk factors of each component (connector) in the architecture as a product of the dynamic complexity (coupling) and the severity level of a failure. The severity level of a failure of component/connector is estimated using FMEA [12].

**Estimate scenario risk factor.** The tool automatically constructs the Markov chain that represents the control flow graph of the active components and connectors in a specific scenario based on the textual representation of the UML sequence diagrams. The scenario risk factor for each severity level is computed using this Markov chain and the estimated values of component/connector risk factors as proposed in [3].

**Estimate use case and overall system risk factors.** The risk factors of each scenario in a specific use case are aggregated to calculate the use case risk factor [3].

---

1 This work is funded in part by grants to West Virginia University Research Corp. from the National Science Foundation Information Technology Research (ITR) Program grant number CCR-0082574, and from the NASA Office of Safety and Mission Assurance (OSMA) Software Assurance Research Program (SARP) managed through the NASA Independent Verification and Validation (IV&V) Facility, Fairmont, West Virginia.
the risk factor for each use case, the tool calculates the overall system risk factor [3].

3. Illustration of the tool outputs

Due to a space limitation we only illustrate some of the output results provided by the ARAT. These results are for the pacemaker [12] which is an implanted device that assists cardiac functions of the heart when the underlying pathologies make the intrinsic heartbeats low. Pacemaker is an example of a critical real-time application because the failure of the software operation of the device can cause loss of a patient’s life. Figure 3 presents the identification of the critical components in the pacemaker example provided by the tool. Thus, the components that have high risk factors with catastrophic severity in multiple scenarios are the most critical components that would require more careful development and/or more testing effort.

The distribution of the overall system risk factor among severity classes is presented in Figure 4. We see that the overall system risk factor is mostly distributed among marginal and catastrophic severity classes which confirms that this is a high risk system.

4. Conclusion and future work

In this paper, we present ARAT, a tool for architectural level risk assessment based on UML specifications. The tool enables early assessment of risk and hence makes it possible for the analyst to identify critical components/connectors or scenarios/use cases early in the software lifecycle. The output of the tool can guide the allocation of development and testing effort based on critical use cases, scenarios, components, and connectors.

Our future plan is to further extend the tool so that it computes static metrics, as well as to collect, store and analyze data which is used for interpretation of quality metrics, even though the result maybe not as sensitive and complete as dynamic metrics for early risk assessment. In addition, we plan to integrate the hazard analysis methodology into our tool to allow automatic and precise estimation of the severity level for each architectural element.

5. References