Information and Software Technology xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Maintaining accurate web usage models using updates from activity diagrams[☆]

Gity Karami[a], Jeff Tian[*,a,b]

[a] Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX 75275, USA
[b] School of Computer Science, Northwestern Polytechnical University, Xi'an, Shaanxi, China

A R T I C L E   I N F O

Keywords:
Usage model
Markov operational profile
Activity diagram
Web application
Maintenance and evolution

A B S T R A C T

Context: Markov operational profile (Markov OP) is a type of usage models for large applications involving state transitions. Such usage models not only help us ensure and maximize product reliability, but can also be used to understand user behavior, and fine-tune system performance and usability. Web usage models can be constructed based on actual usage of the application by target users recorded in existing web logs. Such models constructed before maintenance and evolution may not reflect actual usage of the updated application accurately. At this point, the updated web application has not been deployed yet, so that its actual usage data could not be collected to construct a new Markov OP needed to test the newly updated web application.
Objective: This paper aims at maintaining accurate Markov OP using updates derived from activity diagrams used in web application maintenance and evolution.
Method: Markov OP shares some common characteristics with activity diagrams which describe the application in terms of user activities. We develop a method to update the initial Markov OP by analyzing its differences with the activity diagrams.
Results: We have applied our method in a web application to provide an initial validation of its applicability and effectiveness. After the deployment of the updated web application and new usage data became available, a new Markov OP was constructed. We quantified inaccuracies of the initial Markov OP and the updated Markov OP using the new Markov OP as the reference standard, and quantitatively demonstrated that our method improves the accuracy of the initial Markov OP for the updated web application.
Conclusion: Our new method provides an effective and practical way to maintain accurate Markov OP over web application maintenance and evolution using existing activity diagrams.

1. Introduction

An operational profile (OP) is a usage model that quantitatively characterizes how an application will be used by its target users [19,20]. It consists of a set of operations that the application is designed to perform and their probabilities of occurrence. Several variations of OP based on partitions, tree structures, finite state machines, and Markov chains are commonly used. OP is an effective and practical way to improve reliability and to speed up the development process [17,18]. Using an OP to guide usage-based statistical testing (UBST) ensures that if testing is terminated and the software is shipped because of imperative schedule constraints, the most-used operations will have received the most testing to ensure its reliability.

Web applications provide cross-platform universal access to web resources for the massive user population. With the prevalence of the World Wide Web and its increasing size and complexity, quality assurance for web applications is becoming increasingly important. Markov OP is a good candidate for effective web quality assurance because it captures the usage of web components and related navigations for modern web applications [6,13,15,23]. Markov OP can be constructed for a web application by associating a state to each web page or a group of web pages in a subsite or a functional cluster. Each navigation link or a group of navigation links can be associated with a state transition in the Markov OP. Various log files are routinely kept at web servers to track web usages and problems. Markov OP can be constructed based on these logs.

Accuracy of Markov OP could deteriorate after maintenance and evolution. The initial Markov OP constructed before these changes will not accurately reflect the expected usage for the updated web application, due to addition, deletion, or functional changes to certain web

☆ This work is supported in part by National Science Foundation (NSF) Grant #1126747 and NSF Net-Centric I/UCRC.
* Corresponding author: Department of Computer Science and Engineering, Southern Methodist University, Dallas, TX 75275, USA.
E-mail addresses: gkarami@smu.edu (G. Karami), tian@smu.edu (J. Tian).

https://doi.org/10.1016/j.infsof.2017.11.006
Received 8 February 2017; Received in revised form 8 November 2017; Accepted 10 November 2017
0950-5849/

components resulted from such maintenance activities. At this point, the updated web application has not been deployed yet, so that its actual usage data could not be collected to construct a new Markov OP that is needed to conduct UBST. On the other hand, the user behavior is not expected to change drastically over maintenance and evolution. Therefore, the initial Markov OP could be incrementally updated based on additional information sources and then used in UBST.

One possible information source is activity diagrams commonly used in software development. Activity diagrams describe the application in terms of activities[8]. Such models share some common characteristic with Markov OP. We develop a method to update the initial Markov OP by analyzing its differences with the activity diagrams. We have applied our method in a case study to demonstrate its applicability and effectiveness.

In Section 2, we discuss the related work. In Section 3, we describe the open problem of deteriorating accuracy of Markov OP over maintenance and evolution. In Section 4, we present our method to update the initial Markov OP for the updated web application. In Section 5, we apply our method in a case study to demonstrate its applicability and effectiveness. In Section 6, we discuss the advantages and limitations of our method. Finally, we present our conclusions in Section 7.

2. Related work

In this section, we review related work in Markov OP for web applications and activity diagrams.

2.1. Markov OP, web application, and web logs

Finite state machines (FSMs) are state-based models that can be used for coverage-based testing, e.g., requiring all states and state transitions be traversed [3,22]. For any reasonably sized web application, usage based statistical testing (UBST) can be performed, instead of the less practical and often infeasible coverage of all the states, transitions, and execution sequences of the detailed FSM [15]. Augmented FSM in the form of Markov OP that includes probabilistic usage information supports UBST and reliability analysis [29]. To construct Markov OP, we need to identify information sources and collect data, and then identify states, transitions, input-output relations, and determine usage frequencies of individual transitions. We may need to construct hierarchical Markov OP called Unified Markov Models (UMMs) [15] by expanding some high level states into lower level models to reflect detailed usage of target users.

Actual measurement of usage at customer installations, survey of target customers, and usage estimation based on expert opinion are generic methods for information gathering and OP construction [19]. For existing web applications, the most effective way to obtain usage scenarios and the associated probabilities is through actual measurement of the in-field operations. Log data routinely kept at Web servers represent actual usage, and can be effectively used to construct Markov OP for existing web applications [13,15,27]. Such data have also been used for understanding user behavior, guiding user interface design, exploring user satisfaction, and modeling user activities or tasks [10,11,26,28].

A “hit” is registered in the access logs if a file corresponding to a web page, a document, or other web content is explicitly requested, or if some embedded content is implicitly requested or activated [2]. Most web servers record relevant information about individual accesses in their access logs. Therefore, we would only incur minimal additional cost to use access logs for Markov OP construction. “Requested URL” field in the access log helps us identify all visited web pages and assign each web page or a group of them to a unique state in a Markov OP. State transitions and associated probabilities can be calculated based on “Requested URL” and “Referring URL” fields in the access log.

2.2. Use cases and activity diagrams

Use cases are commonly used in software engineering for specifying functional requirements [30]. A use case describes the way an application is used by its users or entities that interact with the application to achieve their goals. Qualitative usage information in these use cases can be quantified to build usage models such as Markov OP. In use-case-driven software development, the use case model drives all development work from initial gathering of requirements to design, implementation, testing, and deployment.

Activity diagrams, like use cases, describe how a system is used in terms of activities [8]. The activities in the activity diagrams are shown as states that represent the execution of a set of operations. Activity diagrams are employed throughout software development life cycle, including coding, testing, and maintenance [1,9,25]. Activity diagrams are commonly used in the normal software development process. When activity diagrams are used for web application development, each activity can be implemented as one web page or a group of web pages in a subsite or a functional cluster.

Activity diagrams share some common characteristics with Markov OP. Activity diagrams consist of a set of activities and transitions; while Markov OP consists of states, state transitions, and transition probabilities to quantify user behavior and use cases. Therefore, there is a possibility to map a sequence of activities in a activity diagram to a corresponding sequence of states in a Markov OP. This possibility is explored in this paper.

3. Markov OP over maintenance and evolution

Markov OP is constructed based on actual usage of the web application by target users. Maintenance and evolution on web application could affect its functionality, structure, and usage, and leading to deteriorated accuracy of its Markov OP. In this section, we characterize this open problem.

3.1. Maintenance and evolution affect actual usage

There are four types of maintenance and evolution activities: 1) corrective maintenance, 2) perfective maintenance, 3) preventive maintenance, and 4) adaptive maintenance [4,5,14]. Their impact on the actual usage is examined next.

Corrective maintenance refers to modifications necessitated by existing faults in the web application. If we remove some internal faults, actual usage of a web application by target users could change. For example, if there is a broken link in a web application, target users can not reach some specific web pages. A broken link may also cause fault propagation, leading to a group of inaccessible web pages. After fixing the broken link, target users will be able to access the part of the web application which was not accessible previously.

Adaptive maintenance refers to modifications necessary to accommodate a change in the environment, while perfective maintenance improves performance of the web application or its maintainability. Adaptive and perfective maintenance may change actual usage of the web application. For example, adding or removing one or more pages from a web application to cope with new software environment may change actual usage of the web application.

Preventive maintenance, which refers to the modifications necessitated by tolerating potential faults in the web application, also changes the actual usage of the web application. For example, for a frequently visited web page, multiple functionally equivalent web pages from the same initial specifications could be implemented to prevent potential failures. Actual usage of the updated web application would change under this situation.

To summarize, the initial Markov OP constructed earlier would become less accurate for the updated web application after maintenance and evolution. Since Markov OP plays an important role in web