Information and Software Technology 96 (2018) 68–77

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

ARTICLE INFO

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

ABSTRACT

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
Available online 16 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

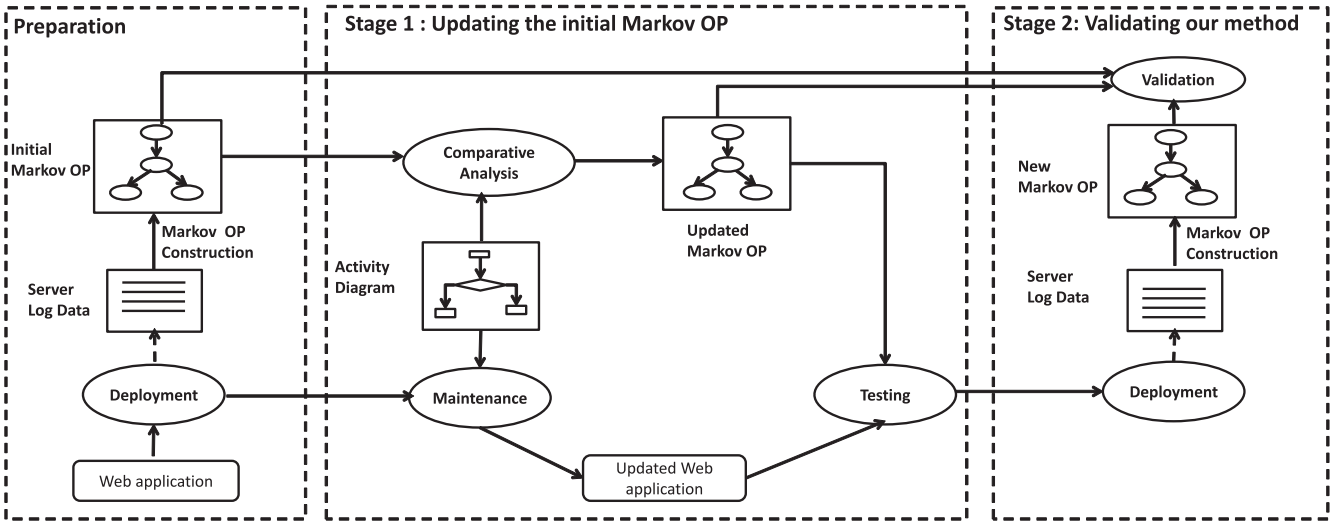

Fig. 1. Our method to maintain accurate Markov OP.

application development and testing, we need to update the Markov OP accordingly to maintain its accuracy. However, we do not have access to the actual measurements, as the updated web application is not under deployment yet. In this situation, survey or expert knowledge could be used to construct another Markov OP. However, those methods only provide less accurate information for Markov OP construction than actual measurement [19]. A new method is needed to address this problem of deteriorating accuracy of Markov OP over maintenance and evolution.

3.2. Possible changes to Markov OP

As we discussed above, if a web application is changed by maintenance and evolution, actual usage of the updated web application also changes. No matter which specific maintenance activity is carried out, the possible changes to the web application and its usage can be classified into three categories:

1. Addition of some components to perform additional functions.
2. Deletion of some components where specific functions are no longer needed.
3. Update to some components to provide somewhat different functions.

Sometimes structural change may lead to functional change. For example, deleting some components may results in migrating part of the functions associated with the deleted components into other components. Similarly, adding some components may result in extracting out functions from pre-existing components for distribution among the newly added components. If we remove or add some components, we may also have functional change in its parent.

The corresponding changes to Markov OP would include both structural changes and changes in transition probabilities related to the following scenarios:

1. Adding a state or state transition
2. Removing a state or state transition
3. Changing functions associated with a state

There will be structural changes in the initial Markov OP due to addition or deletion of a state or a state transition. Structural changes will also affect the probability distribution, because the newly added component will be accompanied by usage not previously accounted for, and the deleted component will no longer be available for usage. In addition, an updated component may also have different usage and may affect the probability distribution.

4. A new method

We develop a new method to maintain accurate Markov OP over maintenance and evolution. The overall approach is described first, followed by detailed description of its individual elements and steps.

4.1. Overall approach

We could take advantage of existing information sources, such as the initial Markov OP and activity diagrams, to maintain accurate Markov OP before actual usage data of the updated web application become available. The initial Markov OP reflects actual usage of target users before maintenance and evolution. We most likely will have the same target users for the updated web application, and their usage patterns are not expected to change drastically. Therefore, the initial Markov OP could be incrementally updated. On the other hand, Markov OP shares common characteristics with activity diagrams which describe the updated web application in terms of activities. By comparing the initial Markov OP and activity diagrams, we could examine how maintenance and evolution affect the usage of the updated web application. The results from this comparative analysis can then be used to update the initial Markov OP.

Fig. 1 illustrates our method for maintaining accurate Markov OP after maintenance and evolution. Our method has two major stages, including:

- Stage 1: Updating the initial Markov OP
- Stage 2: Validating the updated Markov OP

Prior to the above stages, we should have already constructed the initial Markov OP for the web application under deployment. As we described in Section 2, we could extract actual usage of the web application from access logs to construct the initial Markov OP.

In stage 1, we compare the initial Markov OP and activity diagrams to characterize how maintenance and evolution affect actual usage of the updated web application. Using the comparative analysis results and expert knowledge, we construct an updated Markov OP for the updated web application by systematically updating the structure and transition probabilities of the initial Markov OP.

In stage 2, to validate our method, we need to determine whether the updated Markov OP is more accurate than the initial Markov OP.

Beside, we also need to validate our assumption that actual usage do not change drastically over maintenance and evolution. To perform these validation activities, we need to construct a new Markov OP for the updated web application based on actual measurements after its deployment. Then, we compare the initial Markov OP, the updated Markov OP, and the new Markov OP to demonstrate the validity of our method.

Before describing our method in details, we introduce some notations:

- $\langle S, T, P\rangle$: a given Markov OP. We also use $\langle S^I, T^I, P^I\rangle$, $\langle S^U, T^U, P^U\rangle$, and $\langle S^N, T^N, P^N\rangle$ to denote the initial Markov OP, the updated Markov OP, and the new Markov OP respectively.
- $S = \{s_i | i = 1, 2, ...,N\}$; s_i: a state.
- $T = \{t_{ij} | i, j = 1, 2, ...,N\}$; t_{ij} : a transition from s_i to state s_j. We also use n_i to denote the number of possible transitions ($n_i \leq N$) from s_i.
- $P = \{p_{ij} | i, j = 1, 2, ...,N\}$; p_{ij}: the transition probability from state s_i to state s_j, where $0 < p_{ij} \leq 1$ and $\sum_j p_{ij} = 1$.
- $|S|$: cardinality of set S; $|S| = N$.

4.2. Structure update

We utilize the initial Markov OP and activity diagrams to identify all states and state transitions expected to be in the updated Markov OP. Each activity in the activity diagrams represents one web page or a group of web pages in a subsite or a functional cluster. Each state in the initial Markov OP is also associating to one web page or a group of web pages. Activity diagrams have different types of nodes, including activity node, initial node, final node, decision node, merge node, fork node and joint node [8]. To map activity diagrams to Markov OP, we focus on the activity nodes, and consolidate other nodes around activity nodes and related transitions. By comparing the initial Markov OP with the activity diagrams, we could determine how maintenance and evolution affect the structure of the initial Markov OP. We could identify all states and state transitions that should be included in the updated Markov OP corresponding to the updated web application.

In updating the initial Markov OP, we need to consider the following scenarios resulted from web maintenance and evolution: 1) adding a state or a state transition, 2) removing a state or a state transition, and 3) changing functions associated with a state. Fig. 2 illustrates how adding or removing a state transition changes the structure of the initial Markov OP. If a web component is added over maintenance and evolution, it will be represented in the corresponding activity diagram. To maintain the accuracy of the initial Markov OP, we should update it by adding corresponding states and state transitions. For example, in Fig. 2 the initial Markov OP is updated by adding the state j'_4 and the state transition from i' to j'_4.

If some web pages are removed from the web application over maintenance and evolution, we should update the initial Markov OP accordingly. In this situation, the activity diagrams do not have any activity or transition corresponding to those web pages. Since those web pages are no longer accessible to target users, we should update the initial Markov OP by removing the corresponding states and state transitions. For example, in Fig. 2 the initial Markov OP is updated by removing the state j_3 and the state transition from i to j_3.

If a function associated with a specific web page is changed by maintenance and evolution, we do not need to update the structure of the initial Markov OP. However, we may have different usage for the web page, so we may need to update the transition probabilities of the corresponding state transitions in the initial Markov OP.

4.3. Methods for transition probability update

We utilize one of the following methods, in order of preference, to update transition probabilities: 1) partial data estimation method, 2) analogy estimation method, and 3) redistribution estimation method.

Partial data estimation method: If part of the updated web application is under partial deployment, or undergone design inspection, prototype evaluation, or usability testing, we could collect some usage data for the updated subset of the web application. Using such data, we could substitute transition probabilities estimated from the new data for those in the corresponding subset of the initial Markov OP. Fig. 3 illustrates this method. Subset A is updated to become A' and partially deployed, so we have access to partial usage of the updated web application by users. Using the new partial usage data, we calculate transition probabilities of subset A' in the updated Markov OP, while the rest of the updated Markov OP, part B, can be obtained using the corresponding part of the original Markov OP.

Analogy estimation method: If we cannot apply partial data estimation method due to lack of new data, we can utilize the initial Markov OP or another pre-existing Markov OP to update the transition probabilities. In this method, we substitute a subset of the updated Markov OP with a similar subset in the initial Markov OP or in a pre-existing Markov OP. The similarity is established by matching the subset structure and semantics. Some expert judgment is usually involved in establishing this similarity. Fig. 4 illustrates this method. Subset A in the initial Markov OP is updated to become A'. If A' is similar to C in another pre-existing Markov OP in structure and semantics, denoted as $A' \approx C$ in Fig. 4, we would expect similar usage patterns for them. This analogy between subset A' in the updated Markov OP and subset C in another pre-existing Markov OP can be used to substitute C for A' and then integrate with B to update the initial Markov OP. Alternatively, if we can discover subset C from the initial Markov OP, we can use it instead. A special case of analogy estimation method is to utilize functional migration information. If a function supported by a specific component is migrated to another, the corresponding transition

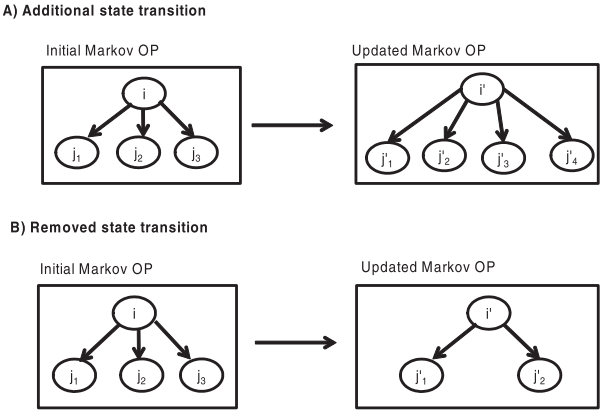

Fig. 2. Structure update.

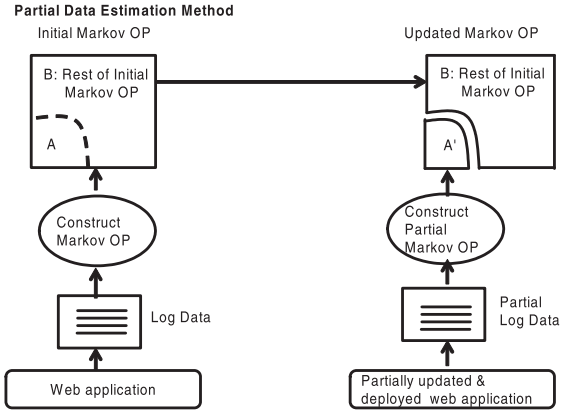

Fig. 3. Partial data estimation method.

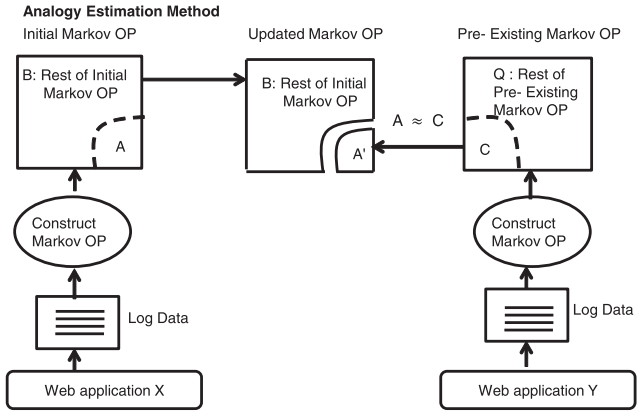

Fig. 4. Analogy estimation method.

probability distribution can be used in its new surrounding environment.

Redistribution estimation method: If we cannot apply partial data or analogy estimation methods due to unavailable data or match, we can redistribute transition probabilities of the initial Markov OP after the addition, deletion, or functional change of certain states and/or transitions following some rules described below.

4.4. Transition probability redistribution

If a state transition is added to the initial Markov OP, we need to estimate the transition probability of the additional state transition. Due to the structural change, we also need to update the transition probabilities of its siblings. In Fig. 2, we not only need to estimate the transition probability for the additional state transition from i' to j_4', but also need to update the transition probabilities to its siblings. If a state transition is removed from the initial Markov OP, we just need to update the transition probabilities of its siblings due to the structural change. In Fig. 2, after removing the state transition from i to j_3, we need to update the transition probabilities from i' to j_1' and from i' to j_2'.

A simple way to redistribute state transition probabilities after addition or deletion of a state transition is to equally redistribute the state transition probabilities within a subset in the updated Markov OP. If t_{ij} was removed from the initial Markov OP, we have $n_i' = n_i - 1$. We can equally redistribute transition probabilities among its n_i' siblings in the updated Markov OP, using the following equation:

$$p_{i'k'} = p_{ik} + \frac{p_{ij}}{n_i'}$$

If $t_{i'j'}$ was added to the initial Markov OP, we have $n_i' = n_i + 1$. We need to estimate transition probability $p_{i'j'}$ for the added state transition using new usage data. If such data are not available, we can estimate $p_{i'j'} = \frac{1}{n_i'}$. Then, we redistribute transition probabilities between its siblings equally, using the following equation:

$$p_{i'k'} = p_{ik} - \frac{p_{i'j'}}{n_i}$$

If we have an additional state transition and its siblings have small transition probabilities, the equal redistribution estimation method may lead to negative probability $p_{i'k'}$ if $p_{ik} < \frac{p_{i'j'}}{n_i}$. To address this issue, we introduce proportional redistribution estimation method.

The proportional redistribution estimation method mimics previous users behavior captured in the initial Markov OP. If t_{ij} was removed from the initial Markov OP, we just need to distribute p_{ij} proportionally between corresponding siblings in the updated Markov OP, using the following equation:

$$p_{i'k'} = p_{ik}*\left(1 + \frac{p_{ij}}{1 - p_{ij}}\right)$$

If $t_{i'j'}$ was added to the initial Markov OP, we first need to estimate $p_{i'j'}$ using the same method described earlier. Then, we need to update transition probability of its siblings proportionally, using the following equation:

$$p_{i'k'} = p_{ik}*\left(1 - p_{i'j'}\right)$$

In this case, we would not run into the same problem of negative transition probabilities for the equal redistribution estimation method.

4.5. Validation

We need to collect actual usage data of the updated web application by target users after its deployment to construct a new Markov OP for the updated web application. We assume the new Markov OP is 100% accurate and consider it as a reference Markov OP or a reference standard. We can quantify inaccuracy level of the initial Markov OP by comparing the initial Markov OP with the new Markov OP. We first identify the following subsets:

- Subset of accurate states in the initial Markov OP S_1^I contains common states which are present in both the initial Markov OP and the new Markov OP.
- Subset of inaccurate states in the initial Markov OP S_0^I contains extra states, missing states, and incorrect states. Extra states are present in the initial Markov OP, but absent in the new Markov OP. Missing states are absent in the initial Markov OP, but present in the new Markov OP. Incorrect states in the initial Markov OP support different functionality rather than their corresponding states in the new Markov OP.

We quantify state inaccuracy of the initial Markov OP σ^I using the following equation:

$$\sigma^I = \frac{|S_0^I|}{|S^N \bigcup S^I|}$$

Let $S^N \bigcup S^I = S$, we then have $\sigma^I = \frac{|S_0^I|}{|S|}$. Notice that $|S| = |S_0^I| + |S_1^I|$, because we partitioned S above into a subset of accurate states S_0^I and a subset of inaccurate states S_1^I.

To quantify probability inaccuracy of the initial Markov OP, we first compare the probability of every transition in the existing Markov OP p_{ij}^I with the probability of corresponding transition in the new Markov OP p_{ij}^N and calculate their absolute difference $\left|p_{ij}^I - p_{ij}^N\right|$. If a corresponding transition is absent, we treat its transition probability as 0 in this calculation. Then, we calculate probability inaccuracy of the initial Markov OP ε^I using the following equation:

$$\varepsilon^I = \frac{\sum_{ij}\left|p_{ij}^I - p_{ij}^N\right|}{K}$$

Where $K = |P^I \bigcup P^N|$, i.e. the size of the union of transitions actually in the models.

Similarly, we can quantify inaccuracy level for the updated Markov OP by comparing the updated Markov OP with the reference Markov OP using the same method above. As discussed earlier, we constructed the updated Markov OP for the updated web application using the initial Markov OP and activity diagrams. We expect the updated Markov OP to be more accurate for the updated web application than the initial Markov OP. The following is our validation question:

- VQ: *The updated Markov OP should be more accurate than the initial Markov OP,* i.e., the former should match the new Markov OP more

closely than the match between the latter and the new Markov OP.

To answer the validation question quantitatively, we need to validate that state inaccuracy of the updated Markov OP σ^U is lower than state inaccuracy of the initial Markov OP σ^I. In addition, we need to validate that probability inaccuracy of the updated Markov OP ε^U is lower than probability inaccuracy of the initial Markov OP ε^I. Therefore, the following expressions should be true to validate our method:

$$\sigma^U < \sigma^I$$

$$\varepsilon^U < \varepsilon^I$$

5. Case study

We used a student payments (SP) website as a case study to provide an initial validation of the applicability and effectiveness of our approach. Predecessor of the SP website was initially developed and deployed in 2013. SP helped international students make payments to register in different exams such as TOEFL and GRE, to reschedule their exams, to report their grades to different universities, and to request a re-score. It also helped international students pay application fees and student visa fees.

We applied our method on the access logs of the SP website from two consecutive months in 2015, one before and one after some maintenance activities. Since individual users didn't usually use the SP website frequently, there wasn't a long term behavior change for each user. Therefore, a relatively short period should be a representative snapshot of the collective usage as long as we had sufficient amount of data for the period. The number of hits in the access logs of the pre and post maintenance SP website were 1484 and 1722 respectively. In this section, we applied our method to the overall usage model of the SP website as well as to a detailed usage model for the "GRESubset".

5.1. Initial Markov OP construction

Fig. 5 shows the initial Markov OP for SP, our high level Markov OP. Fig. 6 shows the initial Markov OP constructed for "GRESubset", one of our low level Markov OP. All the states are labeled and each link is annotated by its corresponding state transition probability. All the external links and associated transition probabilities are explicitly identified, e.g., in Fig. 6, $P_{E_1G} = 0.066$ for the external link from E_1 to the external node G.

To construct an initial Markov OP, we first extracted all visited web pages using "Requested URL" field in the access logs. We assigned one visited web page or a group of visited web pages to a unique state in the initial Markov OP. For example, we assigned the "Default" state in the initial Markov OP corresponding to the "Default.aspx" web page. The group of visited web pages related to "GRE" is represented as a single state in Fig. 5.

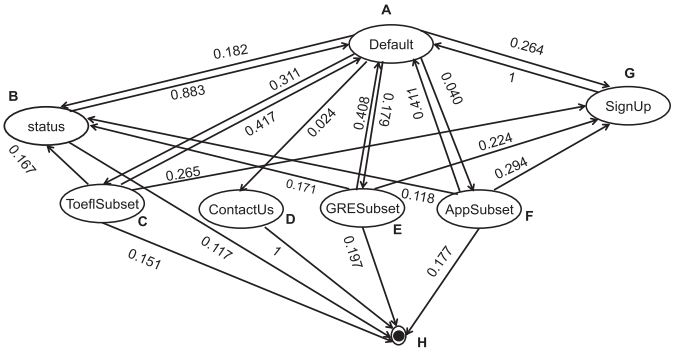

Fig. 5. Initial Markov OP for SP as a whole.

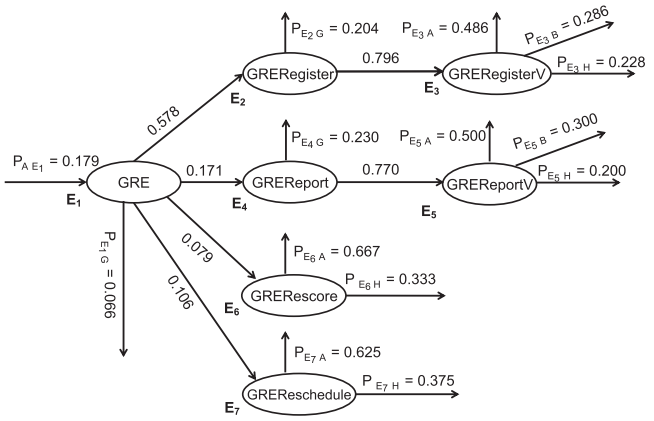

Fig. 6. Initial Markov OP for "GRESubset".

After extracting all possible states in the initial Markov OP, we extracted all incoming navigation links for each web page using "Referring URL" field. We associated one navigation link or a group of navigation links to a unique state transition in the initial Markov OP. For example, we observed that there is a navigation link from "Default.aspx" to "SignUp.aspx", so we assigned a navigation link to the state transition from "Default" to "SignUp" in the initial Markov OP. Similarly, the navigation links from "Default.aspx" to the group of "GRE" related web pages is represented as a single state transition from "Default" to "GRESubset".

After extracting all possible states and state transitions, we calculated the relative frequency of each state transition as its transition probability using the access log recorded before web maintenance was performed.

5.2. Structure update

Next, we analyze the initial Markov OP constructed and compare it with the activity diagram in Fig. 7. As part of the effort to adapt the SP website to new market environment to help users to pay their SEVIS fee and to update the SP website to remove unnecessary services for "GRERescheduler" and "GRERescore", we developed the activity diagram to update the SP website. Each activity in the activity diagram is associated with one web page or a group of web pages. The following are the results of comparing the activity diagram and the initial Markov OP:

- There is no state in the initial Markov OP corresponding to the "SEVISSubset" activity in the activity diagram, and all of the incoming and outgoing state transitions to and from "SEVISSubset" are missing in the initial Markov OP.
- "GRERescore" state and "GRERescheduler" state in the initial Markov

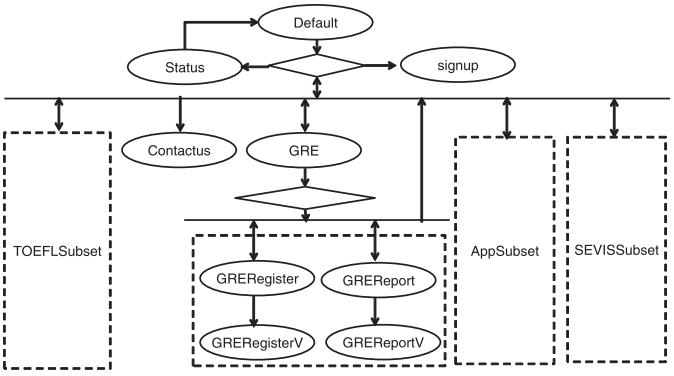

Fig. 7. Activity diagram.

OP for "GRESubset" can not be mapped to any activity in the activity diagram, so these states and all of their incoming and outgoing state transitions are extra.

Therefore, the initial Markov OP did become less accurate over maintenance and evolution. To maintain the accuracy of Markov OP, we updated the initial Markov OP as follows:

- Adding states and state transitions to the initial Markov OP: In this case, we added the "SEVISSubset" state and its incoming and outgoing state transitions.
- Removing states and state transitions from the initial Markov OP: In this case, we removed "GREReschedule" and "GRERescore" states, and all their incoming and outgoing state transitions.
- Functional change: Removing "GREReschedule" and "GRERescore" states and all their incoming and outgoing state transitions will result in functional change in its parent "GRE" state.

Functions associated with all other states in the initial Markov OP did not change over maintenance and evolution.

5.3. Transition probabilities update

Fig. 8 shows the high level Markov OP for the updated web application. For the "SEVISSubset" state and some incoming or outgoing state transitions added to the initial Markov OP, we applied analogy estimation method to estimate their transition probabilities. We utilize "AppSubset" and its transition probabilities in the initial Markov OP to estimate incoming and outgoing state transition probabilities of SEVISSubset", as there is structural similarity between these states and the tasks are similar. For example, we estimate the transition probability from "Default" to "SEVISSubset" in the updated Markov OP to be 0.040, the same as the transition probability from "Default" to "AppSubset" in the initial Markov OP.

Fig. 9 shows the updated Markov OP for "GRESubset". Since state transitions from "GRE" to "GRERescore" and from "GRE" to "GREReschedule" were removed, we need to apply appropriate method to update the probabilities of the sibling transitions from the "GRE" state. As no new data are available at this time, and no analogy can be established, we apply proportional redistribution estimation method to estimate the transition probabilities. Therefore, transition probability from "GRE" to "GRERegister" increases from 0.578 to $0.578*(1 + \frac{0.185}{1-0.185}) = 0.709$, transition probability from "GRE" to "GRERegister" increases from 0.171 to 0.210, and transition probability from "GRE" to "SignUp" increases from 0.066 to 0.081.

Furthermore, we should update the transition probability of the incoming state transition to the "GRE" state, because "GRERescore" and "GREReschedule" functions are no longer available. In the initial Markov OP in Fig. 6, the transition probability from "GRE" to "GREReschedule" is 0.106 and the transition probability from "GRE" to

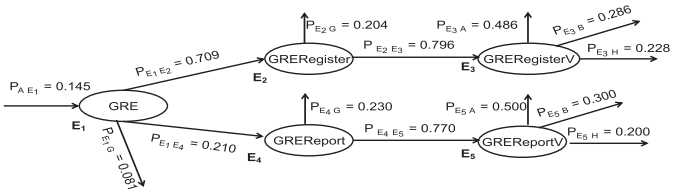

Fig. 9. Updated Markov OP for "GRESubset".

"GRERescore" is 0.079. Since these state transitions are no longer available, we should reduce transition probability to "GRE" to $0.179*(1 - (0.079 + 0.106)) = 0.145$ in Fig. 9. This reduction also needs to be propagated up to its parent Markov OP, with the transition probability from "Default" to "GRESubset" reduced by 0.034 from 0.179 in Fig. 5 to 0.145 in Fig. 8.

For the estimated transition probability of 0.040 for the additional state transition from "Default" to "AppSubset" in Fig. 8, 0.034 can be taken from the above transaction probability reduction from "Default" to "GRESubset". The remaining 0.006 need to be taken from its other siblings. We use proportional redistribution estimation method to estimate transition probabilities of the sibling transitions from the "Default" state. Therefore, we updated transition probability from "Default" to "ToeflSubset" from 0.311 to $0.311*(1 - 0.006) = 0.309$, "Default" to "SignUp" from 0.264 to 0.262, "Default" to "status" from 0.182 to 0.181, and "Default" to "ContactUs" from 0.024 to 0.023.

For all the remaining state transitions in both Figs. 8 and 9, the transition probabilities remained the same as their corresponding part in the initial Markov OP. However, if new data become available for some of these state transitions, the corresponding transition probabilities can be updated accordingly by using our partial data estimation method.

5.4. Validation

We constructed a new Markov OP based on the actual usage of the updated web application by target users using the access log recorded after its deployment to validate our method. Fig. 10 displays the new Markov OP for the updated web application as a whole. Fig. 11 displays the new Markov OP for the updated "GRESubset". The validation question introduced in Section 4 can be examined below.

We can quantify inaccuracy level of a given Markov OP by comparing it with the new Markov OP. By comparing the initial Markov OP and the new Markov OP, we found that the initial Markov OP and the new Markov OP share a majority of common states and state transitions. However, the new Markov OP in Fig. 10 has the "SEVISSubset" state which is not in the initial Markov OP in Fig. 5. In addition, the initial Markov OP for "GRESubset" in Fig. 6 included two extra states "GRERescore" and "GREReschedule" missing from the new Markov OP for "GRESubset" in Fig. 11. The transition probability of every state transition in the initial Markov OP is close to the corresponding

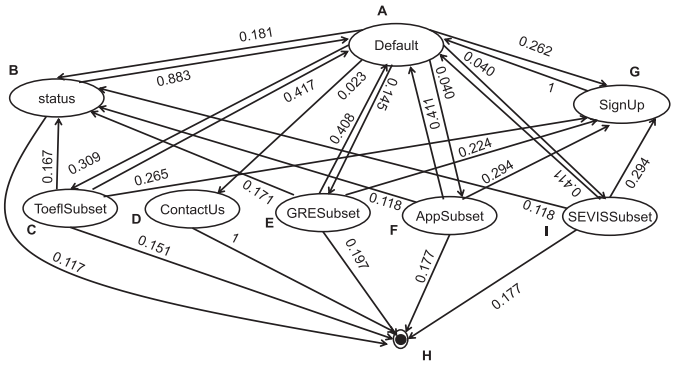

Fig. 8. Updated Markov OP for SP as a whole.

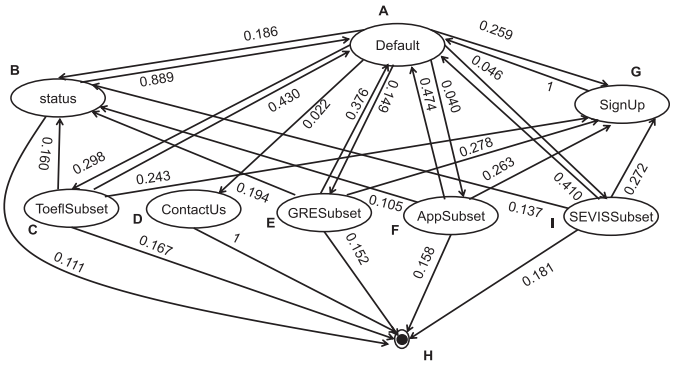

Fig. 10. New Markov OP for SP as a whole.

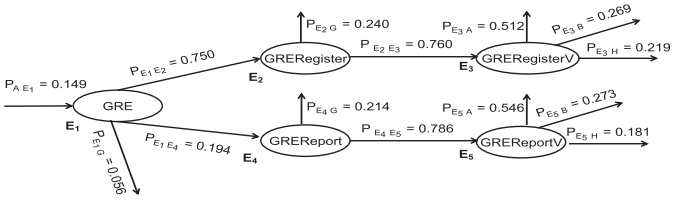

Fig. 11. New Markov OP for "GRESubset".

transition probability in the new Markov OP. In the worst case, the state transition probability from "GRE" to "GRERegister" in the initial Markov OP for "GRESubset" in Fig. 6 is 0.578 while the corresponding state transition probability in the new Markov OP for "GRESubset" in Fig. 11 is 0.750. In the following, we quantify state and probability inaccuracy for the initial Markov OP.

After comparing the initial Markov OP with the new Markov OP, we identified accurate and inaccurate states as:

S_1^I={*Default*, *Status*, *Singup*, *TofelSubset*, *ContactUs*, *AppSubset, GRESubset, H, GRE, GREregister, GREReport, GREReportV, GRERegisterV*}

S_0^I = {*GRERescore*, *GREReschedule*, *SEVISSubset*}

We calculated state inaccuracy of the initial Markov OP σ^I using the following equation:

$$\sigma^I = \frac{|S_0^I|}{|S^N \bigcup S^I|} = \frac{|S_0^I|}{|S|} = \frac{3}{16} = 0.187$$

We also calculated probability inaccuracy of the initial Markov OP. After comparing the probability of every transition in the initial Markov OP p_{ij}^I with the probability of corresponding transition in the new Markov OP p_{ij}^N, we calculated their absolute difference $\left|p_{ij}^I - p_{ij}^N\right|$. Table 1 shows transition probabilities in the initial Markov OP and the new Markov OP and their differences. We calculated probability inaccuracy of the initial Markov OP ε^I using the following equation:

$$\varepsilon^I = \frac{\sum_{i,j}\left|p_{ij}^I - p_{ij}^N\right|}{K} = \frac{4.088}{46} = 0.089$$

Similarly, we quantified the inaccuracy level of the updated Markov OP. After comparing the updated Markov OP and the new Markov OP, we found that they have the same structure. In addition, the transition probability of every state transition in the initial Markov OP is close to the corresponding transition probability in the new Markov OP. In the worst case, state transition from "GRESubset" to "SignUp" in the updated Markov OP in Fig. 8 has transition probability 0.224, while the corresponding state transition probability in the new Markov OP in Fig. 10 is 0.278. In the following, we quantify state and probability inaccuracy for the updated Markov OP.

After comparing the updated Markov OP with the new Markov OP, we identified accurate and inaccurate states as:

S_1^U={*Default*, *Status*, *Singup*, *TofelSubset*, *ContactUs*, *AppSubset, SEVISSubset, GRESubsetm, H, GRE, GREregister, GREReport, GREReportV, GRERegisterV*}

$S_0^U = \emptyset$

We calculated state inaccuracy of the updated Markov OP σ^U using the following equation:

$$\sigma^U = \frac{|S_0^U|}{|S^N \bigcup S^U|} = \frac{|S_0^U|}{|S|} = \frac{0}{14} = 0$$

We also calculated probability inaccuracy of the updated Markov OP. After comparing the probability of every transition in the updated Markov OP p_{ij}^U with the probability of corresponding transition in the new Markov OP p_{ij}^N, we calculated their absolute difference $\left|p_{ij}^U - p_{ij}^N\right|$. Table 2 shows transition probabilities in the updated Markov OP and in the new Markov OP and their differences. We calculated probability inaccuracy of the updated Markov OP ε^U using the following equation:

$$\varepsilon^U = \frac{\sum_{i,j}\left|p_{ij}^U - p_{ij}^N\right|}{K} = \frac{0.756}{40} = 0.019$$

By analyzing the results we found that state inaccuracy of the updated Markov OP $\sigma^U = 0$ is lower than state inaccuracy of the initial Markov OP $\sigma^I = 0.187$. In addition, probability inaccuracy of the updated Markov OP $\varepsilon^U = 0.019$ is lower than probability inaccuracy of the initial Markov OP $\varepsilon^I = 0.089$. Therefore, our VQ is validated, i.e, *the updated Markov OP is more accurate than the initial Markov OP.*

Table 1
Probability inaccuracy of the initial Markov OP.

	p_{ij}^I	p_{ij}^N	$\left\|p_{ij}^I - p_{ij}^N\right\|$
p_{AB}	0.182	0.186	0.004
p_{AC}	0.311	0.298	0.013
p_{AD}	0.024	0.022	0.002
p_{AE}	0.179	0.149	0.030
p_{AF}	0.040	0.040	0
p_{AG}	0.264	0.259	0.005
p_{AI}	0	0.046	0.046
p_{BA}	0.883	0.889	0.006
p_{BH}	0.117	0.111	0.006
p_{CA}	0.417	0.430	0.013
p_{CB}	0.167	0.160	0.007
p_{CG}	0.265	0.243	0.022
p_{CH}	0.151	0.167	0.016
p_{DH}	1	1	0
p_{EA}	0.408	0.376	0.032
p_{EB}	0.171	0.194	0.023
p_{EG}	0.224	0.278	0.054
p_{EH}	0.197	0.152	0.045
p_{GA}	1	1	0
p_{FA}	0.411	0.474	0.063
p_{FB}	0.118	0.105	0.013
p_{FG}	0.294	0.263	0.031
p_{FH}	0.177	0.158	0.019
$p_{E_1E_2}$	0.578	0.750	0.172
$p_{E_1E_4}$	0.171	0.194	0.023
$p_{E_1E_6}$	0.079	0	0.079
$p_{E_1E_7}$	0.106	0	0.106
p_{E_1G}	0.066	0.056	0.010
p_{E_2G}	0.204	0.240	0.036
$p_{E_2E_3}$	0.796	0.760	0.036
p_{E_3A}	0.486	0.512	0.026
p_{E_3B}	0.286	0.269	0.017
p_{E_3H}	0.228	0.219	0.009
p_{E_4G}	0.230	0.214	0.016
$p_{E_4E_5}$	0.770	0.786	0.016
p_{E_5A}	0.500	0.546	0.046
p_{E_5B}	0.300	0.273	0.027
p_{E_5H}	0.200	0.181	0.019
p_{E_6A}	0.667	0	0.667
p_{E_6H}	0.333	0	0.333
p_{E_7A}	0.625	0	0.625
p_{E_7H}	0.375	0	0.375
p_{IG}	0	0.272	0.272
p_{IA}	0	0.410	0.410
p_{IB}	0	0.137	0.137
p_{IH}	0	0.181	0.181
$\sum_{i,j}\left\|p_{ij}^I - p_{ij}^N\right\|$			4.088
ε^I			0.089

Table 2
Probability inaccuracy of the updated Markov OP.

	p_{ij}^U	p_{ij}^N	$\left\|p_{ij}^U - p_{ij}^N\right\|$
p_{AB}	0.181	0.186	0.005
p_{AC}	0.309	0.298	0.011
p_{AD}	0.023	0.022	0.001
p_{AE}	0.145	0.149	0.004
p_{AF}	0.040	0.040	0
p_{AG}	0.262	0.259	0.003
p_{AI}	0.040	0.046	0.006
p_{BA}	0.883	0.889	0.006
p_{BH}	0.117	0.111	0.006
p_{CA}	0.417	0.430	0.013
p_{CB}	0.167	0.160	0.007
p_{CG}	0.265	0.243	0.022
p_{CH}	0.151	0.167	0.016
p_{DH}	1	1	0
p_{EA}	0.408	0.376	0.032
p_{EB}	0.171	0.194	0.023
p_{EG}	0.224	0.278	0.054
p_{EH}	0.197	0.152	0.045
p_{GA}	1	1	0
p_{FA}	0.411	0.474	0.063
p_{FB}	0.118	0.105	0.013
p_{FG}	0.294	0.263	0.031
p_{FH}	0.177	0.158	0.019
$p_{E_1E_2}$	0.709	0.750	0.041
$p_{E_1E_4}$	0.210	0.194	0.016
p_{E_1G}	0.081	0.056	0.025
p_{E_2G}	0.204	0.240	0.036
$p_{E_2E_3}$	0.796	0.760	0.036
p_{E_3A}	0.486	0.512	0.026
p_{E_3B}	0.286	0.269	0.017
p_{E_3H}	0.228	0.219	0.009
p_{E_4G}	0.230	0.214	0.016
$p_{E_4E_5}$	0.770	0.786	0.016
p_{E_5A}	0.500	0.546	0.046
p_{E_5B}	0.300	0.273	0.027
p_{E_5H}	0.200	0.181	0.019
p_{IG}	0.294	0.272	0.022
p_{IA}	0.411	0.410	0.001
p_{IB}	0.118	0.137	0.019
p_{IH}	0.177	0.181	0.004
$\sum_{i,j}\left\|p_{ij}^U - p_{ij}^N\right\|$			0.756
ε^U			0.019

6. Discussion and future work

The initial Markov OP will become less accurate after maintenance and evolution. However, the updated web application under testing has not been deployed yet, so that we do not have actual usage data of the updated web application to construct a more accurate Markov OP needed for UBST. In this research, we provide a new method to address this open problem.

Our method to maintain accuracy of Markov OP is practical, as we utilize the initial Markov OP and activity diagrams with little extra cost. User behavior is not expected to change drastically after maintenance and evolution, so that we could predict usage patterns of the updated web application using the initial Markov OP as the starting point. By comparing the initial Markov OP and the activity diagrams, we could update its structure. The transition probabilities can be updated using partial data estimation method, analogy estimation method, or redistribution estimation method. If we could collect partial new data through partial deployment of the updated web application or its internal inspection, testing, and evaluation activities, we update the transition probabilities of the state transitions using partial data estimation method. If we can find a subset in an existing Markov OP which has similar structure or semantics, we employ analogy estimation method. Otherwise, we employ redistribution estimation method. Our method is initially validated through a case study. The updated Markov OP constructed based on our method is more accurate than the initial Markov OP for the updated web application.

When such updated Markov OP is used to test web applications, it will have the following impact in our case study as demonstrated by a related study [17]:

- Better test coverage: All the newly added components are captured in our updated Markov OP to ensure their test coverage, while such components were missing from the original Markov OP.
- Higher test efficiency: All the removed components are no longer present in our updated Markov OP, so we don't need to waste test effort on them.
- Better quality assurance: The updated Markov OP give us a better focus in testing the web application according to its actual usage, which would help us assure its reliability.

Our method has some limitations. Our initial validation based on a single case study needs to be replicated and scaled up in realistic industrial settings to provide more evidence of the applicability and effectiveness of our method. We map activity diagrams to the initial Markov OP to find out how maintenance and evolution lead to changes in actual usage of the web application. In our case study, there was a simple one-to-one mapping of the states in our Markov OP to the activities in our activity diagram. However, if the activities in the activity diagrams and the states in the Markov OP are associated with groups of web pages in different ways, we may not be able to map the activity diagrams exactly to the initial Markov OP. We plan to use cognitive models [12,21] to support many-to-many mapping between the initial Markov OP and activity diagrams to address this limitation.

Expertise in analogy estimation methods is another limitation of our method. As we discussed in Section 4, we may update transition probabilities of the initial Markov OP based on similarities in structure and semantics as judged by experts. The expert with rich experience and high level of expertise would produce more accurate judgments and estimations than those with limited experience and/or expertise.

Scalability and accuracy of activity diagrams are other limitations for our method. In large scale software development, teams may be distributed over wide geographical areas, and often collaborate asynchronously. Large-scale software development also increases the complexity of activity diagrams [7]. Although it is possible to define such diagrams hierarchically, it will be difficult for designers to follow their relationships and obtain an overall view of the specification. We need to address the difficult issue of mapping such complex activity diagrams to the Markov OPs. We plan to apply our method in some large scale case studies, such as the telecom web site used in [13], and to further validate our method in such realistic industrial settings.

Activity diagrams are commonly used in the normal software development process. However, task models specify more detailed user interaction with the user interface [24]. Task models can potentially lead to better prediction of actual usage of the updated web application. We plan to utilize task models as a replacement to activity diagram in our method to update the initial Markov OP in the future. Some preliminary work in this direction looks promising [16].

Accuracy of the initial Markov OP decreases gradually over maintenance and evolution. When there is a small change in the web application, accuracy of the initial Markov OP does not deteriorate significantly, so that we could still use the initial Markov OP as is. On the other hand, if there is a significant change, the initial Markov OP needs to be updated. Therefore, there exists a break-even point where the initial Markov OP is no longer usable or preferable for the updated web application. Selecting an appropriate break-even point should be addressed in the future.

7. Conclusion

Markov OP is a type of usage models that can support usage based statistical testing (UBST) and help ensure and maximize reliability of web applications. It is also useful to understand user behavior, and to fine-tune system performance and usability. Markov OP can be constructed based on actual usage of the web application by target users. However, accuracy of Markov OP could deteriorate over maintenance and evolution. But we can not collect actual usage data of the updated web application by target users to construct a more accurate Markov OP for use in UBST at the moment, because the updated web application is not under deployment yet. In this paper, we addressed this open problem and provided a novel method for maintaining accuracy of Markov OP over maintenance and evolution.

The main contributions of this paper are: 1) a novel method to maintain the accuracy of Markov OP for web applications over maintenance and evolution, 2) a practical way to update the initial Markov OP using existing information sources, including previously collected usage data, partial new usage data, expert judgment, and activity diagrams used in web application maintenance and evolution, and 3) a case study to provide an initial validation of the applicability and effectiveness of our method.

Since user behavior in the web application is not expected to change drastically over maintenance and evolution, we utilize the initial Markov OP as the starting point to estimate the actual usage of the updated web application by target users. On the other hand, activity diagrams describe a system in terms of activities, and share similar characteristics with Markov OP. Therefore, they were used to characterize how maintenance and evolution affects the actual usage of the updated web application. Furthermore, we utilized the new data collected after maintenance and evolution during such activities as design inspection, prototype evaluation, usability testing, or partial deployment. We provide a novel method to utilize these existing information sources with little extra cost to maintain the accuracy of Markov OP. By comparing the initial Markov OP and the activity diagrams, we could systematically and incrementally update the initial Markov OP to maintain its accuracy.

We applied our method in a web application to provide an initial validation of its applicability and effectiveness. After maintenance and evolution on the web application, we updated the initial Markov OP using the existing information sources. 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. In addition, the updated Markov OP can give us better test coverage, higher test efficiency, and better quality.

To overcome the limitations of our method, we plan to apply and validate it in realistic industrial settings in the future. We also plan to work on reducing the expertise gap to derive consistent and accurate estimates, selecting an appropriate break-even point to apply our method, using cognitive models to address the limitations of mapping Markov OP to complex activity diagrams, and using task models as substitute to activity diagrams in our method to keep Markov OP accurate. After overcoming these limitations, our research will have a significant positive impact on the web application reliability and user experience.

References

[1] E. Arisholm, L.C. Briand, Y. Labiche, The impact of UML documentation on software maintenance: an experimental evaluation, IEEE Trans. Software Eng. 32 (6) (2006) 365–381.
[2] B. Behlendorf, Running a Perfect Web Site With Apache, Second ed., MacMillan Computer, 1996.
[3] B. Beizer, Software Testing Techniques, Second ed., International Thomson Computer Press, 1990.
[4] K.H. Bennett, V.T. Rajlich, Software maintenance and evolution: a roadmap, ICSE '00 Proc. of the Conference on the Future of Software Engineering, (2000), pp. 73–87.
[5] N. Chapin, J.E. Hale, K.M. Khan, J.F. Ramil, W.-G. Tan, Types of software evolution and software maintenance, J. Software Maint. Evol. 13 (2001) 3–30.
[6] L.L. Constantine, L.A.D. Lockwood, Usage-centered engineering for web applications, IEEE Software 19 (2) (2002) 42–50.
[7] S.C.B. de Souza, N. Anquetil, K.M. de Oliveira, A study of the documentation essential to software maintenance, Proc. of the 23rd Annual International Conference on Design of Communication, (2005), pp. 68–75.
[8] R. Eshuis, Symbolic model checking of UML activity diagrams, IEEE Trans. Software Eng. 15 (1) (2006) 1–38.
[9] X. Fan, J. Shu, L. Liu, Q. Liang, Test case generation from UML subactivity and activity diagram, Second Int. Symp. Electron. Commerce Secur. 2 (2009) 244–248.
[10] N. Harrati, I. Bouchrika, A. Tari, A. Ladjailia, Exploring user satisfaction for e-learning systems via usage-based metrics and system usability scale analysis, Comput. Human Behav. 61 (2016) 463–471.
[11] D.M. Hilbert, D.F. Redmiles, An approach to large-scale collection of application usage data over the Internet, 20th Int. Conf. on Software Engineering, Kyoto, Japan, 1998, pp. 136–145.
[12] B.C. Hungerford, A.R. Hevner, R.W. Collins, Reviewing software diagrams: a cognitive study, IEEE Trans. Software Eng. 30 (2) (2004) 82–96.
[13] W. Jaffal, J. Tian, Practical risk-based technique to improve reliability for incremental web application development, Proc. of 27th Int. Conf. on Computer Applications in Industry and Engineering (CAINE-2014), (2014).
[14] H. Kagdi, M.L. Collard, J.I. Maletic, A survey and taxonomy of approaches for mining software repositories in the context of software evolution, J. Software Maint. Evol. 19 (2007) 77–131.
[15] C. Kallepalli, J. Tian, Measuring and modeling usage and reliability for statistical web testing, IEEE Trans. Software Eng. 27 (11) (2001) 1023–1036.
[16] G. Karami, J. Tian, Using task models to maintain accuracy of web usage models, 30th International Conference on Computer Applications in Industry and Engineering (CAINE 2017), (2017).
[17] G. Karami, J. Tian, Improving Web Application Reliability and Testing Using Accurate Usage Models, Studies in Computational Intelligence (SCI) 722 Springer, 2018.
[18] M.R. Lyu, Handbook of Software Reliability Engineering, McGraw-Hill, Inc., 1996.
[19] J.D. Musa, Software Reliability Engineering, McGraw-Hill, 1998.
[20] O.E. Oruc, C. Tatar, An investigation of factors that affect internet banking usage based on structural equation modeling, Comput. Human Behav. 66 (2017) 232–235.
[21] D. Peebles, A.L. Cox, Modelling interactive behaviour with a rational cognitive architecture, Human Comput. Interact. (2009) 1154–1172.
[22] M. Pezze, M. Young, Software Testing and Analysis: Process, Principles and Techniques, first ed., Wiley, 2007.
[23] C. Ramesh, K.V.C. Rao, A. Goverdhan, A semantically enriched web usage based recommendation model, Int. J. Comput. Sci. Inf. Technol. 3 (5) (2011).
[24] F.E. Ritter, G.D. Baxter, E.F. Churchill, Foundations for Designing User-Centered Systems: What System Designers Need to Know about People, Springer Publishing Company, 2014.
[25] S. Sulistyo, A. Prinz, Recursive modeling for completed code generation, Proc. of the 1st Workshop on Behaviour Modelling in Model-Driven Architecture, (2009).
[26] L. Tauscher, S. Greenberg, Revisitation patterns in World Wide Web navigation, Proc. of the ACM SIGCHI Conference on Human Factors in Computing Systems, (1997), pp. 399–406.
[27] J. Tian, S. Rudraraju, Z. Li, Evaluating web software reliability based on workload and failure data extracted from server logs, IEEE Trans. Software Eng. 30 (11) (2004) 754–769.
[28] V.C. Trinh, A.J. Gonzalez, Discovering contexts from observed human performance, IEEE Trans. Human-Mach. Syst. 43 (4) (2013) 359–370.
[29] J.A. Whittaker, M.G. Thomason, A Markov chain model for statistical software testing, IEEE Trans. Software Eng. 20 (10) (1994) 812–824.
[30] K. Wiegers, J. Beatty, Software Requirements, Microsoft Press, 2013.