Profilage des utilisateurs d'applications web avec l'infrastructure de traçabilité et testabilité logicielles SoftScanner

Stage proposé en collaboration entre le laboratoire LIRMM (http://www.lirmm.fr/) et l'entreprise Berger Levrault (https://www.berger-levrault.com/fr/)

Encadrants:

- Abdelhak-Djamel Seriai, MCF Dept Info, FDS, UM, http://www.lirmm.fr/~seriai/, seriai@lirmm.fr
- Bachar Rima, Ingénieur d'études au LIRMM, équipe MAREL, et vacataire à la FDS, UM, rima@lirmm.fr

Contexte

Dans les grandes entreprises, la majorité du budget des logiciels est consacré à la maintenance et l'évolution des logiciels existants plutôt qu'au développement de nouveaux logiciels, ce dernier étant assez coûtant et risqué. Parmi les activités de maintenance et d'évolution logicielles, on note le débogage, l'optimisation, les tests, le profilage des utilisateurs, la sécurité, etc. La mise en œuvre de ces activités peut être réalisée par divers stratégies, dont l'exploitation des traces (*i.e., les logs*), devenant de plus en plus adoptée et systématisée dans le monde du génie logiciel.

Les logs sont parfois les seules ressources disponibles pour la compréhension effective d'un système complexe à grand échelle. Assurer la qualité des informations y incluses est donc d'une importance primordiale, impactant par la suite les décisions prises par les développeurs pour la maintenance/évolution de leur système étudié. De plus, vu que les logs sont générés par des instructions de traçage (*Log Printing Statements (LPSes)*) instrumentant le code source du système étudié, assurer la qualité de ses dernières est également indispensable.

Plusieurs solutions systématiques ont été déjà proposées pour la gestion des logs générés. Cependant, l'instrumentation du code source avec des instructions de traçage demeure coûteuse et sujette à l'erreur humaine, étant ad-hoc, majoritairement manuelle, et surtout dépendante de l'expertise des développeurs qui s'en chargent.

Pour pallier ce problème, le projet SoftScanner, une collaboration entre le LIRMM et l'entreprise Berger-Levrault (BL), a été lancé comme infrastructure dédiée à l'automatisation partielle de l'instrumentation de code source avec des LPSes, selon une/plusieurs objectifs de traçage.

L'infrastructure de SoftScanner est constituée d'un ensemble de petits services, appelés microservices, servant chacun comme un mini-projet et interagissant les uns avec les autres par le biais d'APIs REST. Un client interagit ainsi avec SoftScanner en fournissant un projet à analyser et en choisissant les objectifs de traçage désirés, pour ensuite lancer le processus d'instrumentation. La requête est traitée par un sous-ensemble des microservices qui s'échangent les résultats de leurs traitements entre eux en JSON par le biais de leurs APIs REST. À la fin, un lien vers le projet instrumenté est retourné au client qui pourra ainsi le télécharger.

L'un des microservices de SoftScanner possède comme objectif l'exploration de l'interface graphique (GUI) d'une application web donnée et la construction de sa machine à états. Le microservice a été implémenté pour les applications SPA (Single-Page Applications) créées par Angular.

Objectif et étapes

L'objectif de ce projet TER M1 est de créer un microservice dont le rôle est de fournir un mécanisme de profilage des utilisateurs d'une application Angular instrumentée avec SoftScanner. Il est structuré en 4 tâches obligatoires et 3 tâches bonus :

1) Étude et compréhension du microservice d'exploration des GUIs d'applications web Angular.

2) Réalisation d'une étude bibliographique liée au profilage des utilisateurs, et en particulier l'exploitation des logs et éventuellement d'autres modèles de l'application ciblé (machine à états, AST, etc.) pour implémenter ce profilage.

3) En cas d'absence de logs existants, mise en place d'un système de génération de logs pour une application donné, qui simulera l'utilisation de l'application instrumentée par un ensemble aléatoire d'utilisateurs générés, chacun exécutant un ensemble de scénarios d'exécutions aléatoires à définir (e.g., utilisation de Selenium pour automatiser les scénarios d'exécution via le browser)

4) Conception, implémentation, et test d'une approche permettant le profilage des utilisateurs en utilisant :

- Les logs existants ou les logs générés par le générateur de logs implémenté ;
- Des modèles statiques/dynamiques de l'application s'avérant utiles pour plus de contexte (e.g., machine à états de l'interface graphique de l'application).

5) (Bonus) Création d'une API REST permettant de consommer le service de profilage créé.

6) (Bonus) Création d'une interface graphique web du service de profilage créé.

7) (Bonus) Exploration et création d'une base de données à base de graphes pour le stockage des machines à états des interfaces graphiques d'applications web Angular.

Pourquoi choisir ce projet :

- Acquérir des compétences certaines en développement logiciel de manière générale (principes de design SOLID, patrons de conception, styles architecturaux, etc.).
- Découvrir plusieurs thématiques dans le monde du développement logiciel : la rétro ingénierie (*reverse engineering*), l'analyse du code source, etc.
- Découvrir des techniques de *machine learning* : clustering, topic modeling, etc.
- Acquérir des compétences concrètes en analyse dynamique et tests de logiciels, notamment avec Selenium.
- Avoir une collaboration avec le monde des entreprises (l'entreprise BL, un des grands éditeurs logiciels en France).
- Éventuellement acquérir des compétences en programmation distribuée et la construction d'APIs REST.
- Éventuellement acquérir des compétences en développement web (Spring, Angular, React, etc.).
- Etc.

Si vous avez des questions, contactez Bachar Rima rima@lirmm.fr ou Abdelhak-Djamel Seriai seriai@lirmm.fr

Références Bibliographiques

1) Chen, B. (2020). Improving the Logging Practices in DevOps. https://yorkspace.library.yorku.ca/xmlui/handle/10315/37997
2) Li et al. (2020). A Qualitative Study of the Benefits and Costs of Logging from Developers' Perspectives. IEEE Transactions on Software Engineering, 1–1. https://doi.org/10.1109/TSE.2020.2970422
3) Memon et al. 2003. GUI Ripping: Reverse Engineering of Graphical User Interfaces for Testing. In Proceedings of the 10th Working Conference on Reverse Engineering (WCRE '03). IEEE Computer Society, USA, 260.
4) Domínguez Osorio, J. (2019). Automated GUI ripping for web applications. Uniandes.
5) Eke et al. (2019). A Survey of User Profiling: State-of-the-Art, Challenges, and Solutions. IEEE Access, 7, 144907-144924.
6) Angular : Développez vos applications web avec le framework JavaScript de Google. Daniel Djordjevic, Sébastien Ollivier, William Klein. Editions ENI. Août 2017.