Novosibirsk State Technical University NETI
Pr. K. Marksa, 20
Novosibirsk, Russia, 630073
+7 (383) 346 50 01
rector@nstu.ru

To Prof. Martin Gaedke
Technische Universität Chemnitz
Fakultät für Informatik
Professur Verteilte und Selbstorganisierende Rechnersysteme
Straße der Nationen 62, Haus B
09111 Chemnitz
Saxony, Germany

Review of Sebastian Heil's Doctoral Dissertation

The dissertation of Sebastian Heil, M.Sc., which he developed as a doctoral student with the Technical University of Chemnitz under the supervision of Prof. Dr.-Ing. Martin Gaedke, is dedicated to Web Migration. The main focus of the work is the development of several solutions covering the gaps in the current domain knowledge and practice, so that the risk and effort in the overall migration process are minimized.

The background of the research is well-established with the specified requirements and the review of the state of the art (Chapters 2 and 3). I would however expect to see more theory on software migration in general, e.g. the translation of working code from one programming language into another. Also, I think it would be fair to mention the limitations and disadvantages of the web platform, while currently only the benefits are highlighted. In the requirements analysis for the dissertation's research problem, I think that the "testing migration" aspect is missing, as mostly code migration is considered. This is rather my personal conviction, but I believe that since testing and debugging take up the most development effort, software migration should focus more on using the old code for controlling the quality of the new code. However, what I whole-heartedly agree with is the observation (in p. 64) that the approaches towards web migration that are currently prevalent in academia do not consider the user interaction aspect enough.

All in all, the research objectives established in Chapter 4 do follow logically from the previous text, so I think the author does a good job shaping his area of research. I'd like however to comment on the concept map presented in Figure 3.4 and the following description of the shortcomings of existing approaches:

- G1: Making good software is complex, and reuse of code or design can increase productivity by about 35%. So, I wouldn't expect migration to have much lower requirements towards resources and expertise compared to general software engineering. The goal should rather be measuring and increasing the productivity boost emerging due to the available legacy product.
- G2: It remained unclear to me why existing knowledge preservation is mixed into the same shortcoming as lack of demonstration of desirability and risk management. The former is not as much about risk, but about reuse of design instead of working from scratch. Indeed, in Table 4.1 this issue becomes a separate question as HMW1, and relates to a dedicated research objective RO1.
- G3: for the sake of truth, there has been quite an influential project SUPPLE, which was capable of generating UIs while considering the similarity with the previous UI versions as a component in the cost function. In that, SUPPLE did use a similarity metric, although a quite unsophisticated one, based mostly on Material and not considering Layout.

The main part of the dissertation, which details the original research performed by the author, includes Chapters 5-7, dedicated to crowdsourcing-based Reverse Engineering, the Rapid Web Migration Prototyping, and the controlling of UI similarity. In my opinion, the Rapid Prototyping is arguably the most solid contribution, while the UI similarity is the most promising one. The crowdsourcing used for knowledge extraction seems to be the most controversial part to me, which I will elaborate further. The rest of my review is mostly focused on these chapters of the dissertation, as they present the original research carried out by the author.

The actual experiment performed in Chapter 5 is well documented and has convincing statistical outcome. But the two major doubts I have about the crowdsourcing-based Reverse Engineering are related to the construct validity and the principal utility of the approach. Initially, Concept Assignment is a step in the knowledge (re)discovery (p. 94: "To achieve proper knowledge management, this extraction requires codification…"), which is all right. However, at some point this step seems to replace the whole activity (p. 95: "the AWSM Reverse Engineering method is designed as Concept Assignment method"), without much justification. The discussion of the construct validity of the study (p. 124) doesn't consider this issue either. Interestingly, in the concluding part of the dissertation (p. 220) the author does make the conceptually irrecusable statement: "Crowdsourcing has shown to be effective for the Reverse Engineering activity of Concept Assignment. An open research question is the application of Crowdsourcing in other Reverse Engineering areas."

The above is closely linked to the uncertain practical utility of the approach described in Chapter 5, where a high-qualification task (program understanding) is assigned to a crowd of unskilled workers. In software engineering, it is well known that the very programmer who is going to implement a task is the one who should provide estimate for it, particularly because doing this encourages the task's analysis and understanding. I could not find a justification that in software migration the classified code fragments with common-sense textual descriptions (as shown in Fig. 5.16) would even out to a programmer's knowledge of the system. The achieved accuracy (7/10 code fragments correctly classified) is reasonable, and it might be true that the involved expenses of 20 USD are less than the programmer's time would cost. But in the latter case the output is the programmer who has studied the code in the project, whereas the knowledge increment in crowdworkers who has completed the task is basically wasted. The observation reported in p. 123 "several crowd workers looked up the sample on the Web and also read the surrounding parts from an external source in order to classify" also hints that breaking the code into unrelated microtasks is disadvantageous even for classification, let alone knowledge discovery. This is also something that should have been probably mentioned in the discussion of external validity (p. 125) – how well would the experimental results transfer to less obvious code fragments and more sophisticated concepts representing more complex knowledge? All in all, I believe that the rigor and utility of the approach presented in Chapter 5 are not apparent, even though technically the experiment was performed well and the results suggest that concept assignment, for what it's worth, can be indeed performed with crowdworkers.

The Chapter 6 is dedicated to Rapid Web Migration Prototyping, mostly being focused on automated creation of web UI prototypes similar to the legacy UIs. I believe it addresses an important problem – making prototyping easier encourages its wider use by software engineers, which is certainly beneficial – and makes a solid and well-evaluated contribution. The three experiments undertaken in the Experimental Evaluation part cover the representatives of both developers and end users. With regard to the needs of the former group, the effectiveness, efficiency and understandability of the implemented automated prototype migration workflow are demonstrated, with the use of statistical

methods. Ultimately, the quality of the automatically generated web UI prototypes is verified with representatives of users, who agree that the degree of similarity to the legacy UIs is indeed high.

My critical remarks related to this chapter are rather minor. Again, I see some shift in construct validity: the chapter starts and ends with desirability and feasibility being the objectives (e.g. p. 169), but in the experiments other concepts are used, particularly in the survey questions. Although I agree that it is warranted, I would like to see more explanation of this detailing, particularly for desirability. Further, the author rightfully says that the perception of similarity in users is complex, but does not provide much justification of the related questions (Appendix H.1). The questionnaires as such and their analysis also have some minor issues: e.g. in H.1 Q7 says "A and B are identical", but in Fig. 6.21 it says "layouts identical". The analysis of some of the questions presented in Appendix E, the quality-related ones, is skipped in the respective sub-chapter. Also, the questions are not always on the same level of generality (Q7 in H.1 seems to rather relate to the overall similarity perception), but the replies are just averaged. Despite this, I believe the presented results are significant, and the software implemented to support the web UI prototypes generation is very sound and can see good practical use in Web Engineering.

Finally, in the Chapter 7 the author addresses the management of similarity in new versions of UI, with application in migration from legacy software to the web. I consider this to be a very important issue, somehow neglected in today's UI design, which leads to the infamous users' complaint "each new website design is worse than an old one". The approaches and tools mentioned in the comprehensive state of the art review are indeed short of the required functionality, so the new methodology is called for. The *Task – Behavior – Thesaurus – Layout – Material* framework is a particularly promising advancement, which should be promoted more.

Again, I only have some minor critical points for this chapter. First, the term "Customer Impact Control" appears to me not being entirely accurate, as in UI design customers and users are considered to be distinct groups of stakeholders. The UI similarity discussed in the dissertation is rather aimed on the users. Next, it is not entirely clear why the author only employed the manually created web UIs in the experimental evaluation (p. 190). It would probably make sense to cover some of the automatically generated prototypes from Chapter 6 and see how well the methodology works for them. Also, I believe the data for the tables 7.1 and 7.2 (UI elements and the differences) was obtained through manual labeling of the UIs, which should have required considerable work effort. This is compensated though by the possibility to use trained similarity functions for the purposes of real projects. Overall, I find the results presented in Chapter 7 to be original and convincing and to deserve further development.

The evaluation and the conclusion chapters (8 and 9) neatly sum up the content of the dissertations. I would only like to note that the statement that AWSM "meant to be used in conjunction with one of the complete Web Migration approaches as suitable for the concrete migration situation and environment" (p.207) could have been illustrated with application examples.

The research findings presented in the dissertation have been extensively published, including in the proceedings of A-level international conference (WISE 2016) and the leading international journals. A notable share of the works has been co-authored within the framework of international cooperation, which further reinforces the importance of the results. The writing style is thoroughgoing, there are very few mistakes and the language is good and easy to read. I also believe that the developed software tools to facilitate the web migration process are significant for Web Engineering practice. Despite the certain shortcomings noted in my review, overall I find that the dissertation is an original and substantial research work.

I confirm the theses statements and I recommend the acceptance of the PhD thesis of Sebastian Heil, [redacted]

Maxim Bakaev, PhD, Associate Prof.,
Novosibirsk State Technical University, Russia

15 May 2020