# External Examiner's Report

Christina von Flach Garcia Chavez

This report presents the assessment of the Dissertation entitled ***Supporting a Developer's Discovery of Task-Relevant Information***, authored by **Mr. Arthur de Sousa Marques** and submitted to the Faculty of Science (Computer Science) of the University of British Columbia.

## 1 Summary

**Research Context, Scope and Goals.** The Dissertation of Mr. Arthur de Sousa Marques references the field of Software Engineering, more specifically software development tasks and related challenges under the perspective of software developers.

Software developers seeks information which may typically exists across different kinds of artifacts (bug reports, discussion messages, tutorials) to aid in the completion of a task. Such artifacts include unstructured natural language texts organized in different ways. Given the large amount of information sources and the limited time developers have to spend on any task, there is a need for techniques to support the identification of portions of relevant text for task completion from one or more artifacts. Locating such relevant text can be also time-consuming. Related work includes tool-supported, *task-oriented*, semantic-based approaches however constrained to artifact-specific data or meta-data, which may be difficult to align with developer's daily practice, deploy and evolve (the nature of software development tasks and related artifact types evolve too). Such constraints apply to AnswerBot, the state-of-the-art summarization technique that applies to Stack Overflow posts.

The research goal (aim) stated in Mr. Arthur's dissertation (Chapter 1) is to overcome those constraints and provide *text relevant to a developer's task*

at hand, automatically *extracted from pertinent natural language artifacts by a generalizable technique*, that is, an evolvable, artifact-independent approach that can assist developers in discovering task-relevant information that may be found in different kinds of natural language artifacts.

The research provides a brief and objective literature review on the background concepts, so that the necessary definitions supported by proper references are self-contained in the dissertation text. The research also skims through related knowledge areas such as foraging information theory and natural language processing to present the necessary multidisciplinary research background.

**Methodological and Technological Soundness.** The methods used are mostly suitably described, relevant to the research question(s), and employed appropriately.

Chapter 3 presents a first empirical study (the *formative study*) with the goal of characterizing task-relevant information. The research questions and methods are described. The main concept is that of a Highlight Unit (HU), a full sentence containing any highlight by a participant.

The main results from the analysis of highlights produced by developers indicate that between 1% to 20% of the text of an artifact was considered relevant to a task. The researchers also observed consistency in the meaning of the relevant text as captured using frame semantics, suggesting that semantic-based approaches may be more appropriate for the automatic identification of task-relevant text. These techniques are based on semantic patterns that arise from the empirical analysis of the text relevant to a task in multiple kinds of artifacts, and incorporate the semantics of words and sentences to identify relevant text to a developer's task automatically. Mr. Arthur is the first author of a full paper (Characterizing Task-Relevant Information in Natural Language Software Artifacts) with three co-authors published at the IEEE International Conference on Software Maintenance and Evolution (ICSME) in 2020.

Chapter 4 describes the groundwork for producing the corpus called DSandroid used to evaluate the semantic-based techniques detailed in Chapter 5. It describes the selection of tasks, and artifacts pertinent to each task, as well as how three human annotators identified relevant text in each of the artifacts gathered.

Chapter 5 details the semantic-based techniques under investigation for

automatically identifying task-relevant text. The candidate applies the concept of semantics frames to some of the six semantic-base techniques that *incorporate the semantics of words [..] and sentences [..]* to automatically identify text *likely* relevant to a developer's task and empirically assessed them on a corpus of curated tasks and artifacts In his research, Mr. Arthur evaluates the proposed techniques assessing the extent to which they identify text that developers consider relevant in different kinds of artifacts associated with Android development tasks. Preliminary findings reveal that *semantic-based techniques achieve recall comparable to a state-of-the-art technique aimed at one type of artifact [..] while supporting multiple artifact types.*

Mr. Arthur is the first author of a full paper *Assessing Semantic Frames to Support Program Comprehension Activities* published with two co-authors published at the IEEE/ACM 29th International Conference on Program Comprehension (ICPC), 2021. The paper presents the SEFrame tool used in Chapter 5 and discussed in Chapter 7.

Mr. Arthur is the first author of a full paper *Evaluating the Use of Semantics for Identifying Task-relevant Textual Information* published with his thesis supervisor, in the IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2022. The paper brings together the research described in Chapters 4 and 5.

Chapter 6 describes the investigation, through a controlled experiment, *whether a tool called TARTI aids developers thus addressing whether developers can effectively complete a software task when provided with task-relevant text extracted from natural language artifacts.* The version of TARTI used in the experiment embeds the most promising semantic-based technique, BERT with no extension for words or sentences. The tool's accuracy was evaluated by developers in the context of Python implementation tasks.

The technical soundness in using of existing techniques and tools, as well as the novel combinations involving new and existing approaches is convincingly presented and illustrated by means of implemented tools used by software developers. The background information, research results and discussion are clearly reported and facilitate the overall positive impression of technical soundness.

**Contributions to Knowledge.** The research has been partially published on top software engineering conferences with peer review committees com-

posed by experts in one or more research topics related to the Dissertation. The dissertation critically analyzes the relevant related work in the software engineering field and states the novelty in (or unawareness of) the usage of the meaning of text in techniques that assist developers in discovering task-relevant text over different types of natural language artifacts.

Based on such information, we may say that the research brings contributions to knowledge, among them:

- results from the analysis of highlights produced by developers indicate that between 1% to 20% of the text of an artifact was considered relevant to a task.
- the automatic identification of task-relevant text with semantic-based techniques can support generalizability to several kinds of natural language artifacts;
- the automatic identification of relevant text for a task from multiple natural language artifacts can perform as good as the AnswerBot, the state-of-the-art tool specific to Stack Overflow Q&A format and metadata;
- corpora of curated task-specific relevant texts, extracted from different kinds of natural language artifacts, can be described and built for the use and evaluation of discovery strategies and tools;
- under the subjectivity of *relevance*, discussions speculate whether the explicit need for generalizability may overcome the concern with the kinds of artifacts towards the kinds of strategies used by developers;
- The TARTI tool let us know that it is feasible to automate the provision of task-relevant text assists developers in effectively completing a software development tasks in simple domains and may foster the interest in exploring more complex domains.
- The dissertation created three data sets that can be used in future work or by researchers that investigate problems in the same domain. The dissertation also describes the planning and detailed procedures for the construction of the Android (development) Task corpus.

**Impact on the Software Engineering discipline.** The Dissertation topic is relevant to the state-of-the-art and the state-of-the-practice in the Software Engineering field. Its main contribution is related to the preliminary results that show that semantic-based techniques perform equivalently well across multiple artifact types and that a tool that automates the provision of task-relevant text may assist developers in effectively completing a software development task.

This finding may bring new opportunities for future research that may result in advances for software engineering in general – theory and practice.

**Related Work, implications and limitations of the research.** The analyses and conclusions drawn from the research are well-justified and integrated into the larger field of knowledge, and contrasted to the main state-of-the-art tool/approach.

In the chapters that present proposal or empirical studies, the implications and limitations of the research are discussed. Moreover, threats to the validity of the results and actions to mitigate them are presented.

**Writing of the Dissertation document.** The Dissertation's Preface describes Mr. Arthur's contributions to published papers and other collaborative work developed with several co-authors. Mr. Arthur is the first author of four peer-reviewed papers related to his dissertation, published in high-quality Software Engineering conferences such as ESEC/FSE 2019, ICSME 2020, ICPC 2021 and SANER 2022. The participation of collaborators is clearly stated in the Preface.

The Dissertation is well-structured and clearly presented throughout eight Chapters: introduction, related work, first controlled experiment, corpus, algorithms, tool evaluation, discussion and conclusion. Chapter 1 (*Introduction*) clearly states the thesis theme and context, provides a good and very helpful illustrative example, a brief description of the state of the art, the thesis statement, and states the thesis contributions. Chapter 2 presents a review of relevant literature: it provides sufficient background information to enable a non-specialist scholar to understand them and their relation to the research contributions.

In the remaining Chapters (3–8), the candidate demonstrates his ability to perform analyses and draw well-justified conclusions from the research, understand the implications of his research, contrasted with related work

and integrated into the larger software engineering field, and reflecting on the contributions of the present work and opportunities for future work.

# 2 Recommended Revisions

In general, dissertations that try to adapt the text of previously published work to their Chapters, lack comprehensive introduction and conclusions. In Chapter 1, I missed an explicit link between general research goals research questions, research methods applied and related results that are detailed in Chapters 3, 5 and 6 – possibly a script or roadmap for the thesis in the introduction that glues together the procedures and outcomes of each Chapter to tell the overall story and shows a "big picture" of the research.

Overall, the language is comprehensive and coherent while inaccuracies and minor typos are rarely found. The internal structure and context of the Chapters are adequate with the following exceptions.

In Chapter 3, I recommend improving the preamble of Chapter 3. The mention to "open the questions" followed by "research questions" in the next paragraph hinders comprehension, among other problems. I missed a more careful description of the experimental design presented in Chapter 3. What is/are the goal(s) of the experimental study? Finally, I recommend rewriting the rationale for RQ1 or split the question in two.

*RQ1: How much agreement is there between developers about the text relevant to a task? With this question, we seek to understand the amount of information sought for task completion and whether individuals see the same text as relevant to a task.*

# 3 Overall Recommendation

Mr. Arthur showed in his Dissertation that the problem of finding task-relevant information from one or more artifacts is relevant for SE. Moreover, he was able to critically analyze the related work and position his research and its findings within the broader field of SE.

He used and described in detail the appropriate methodology to guide the execution and evaluation of the research work undertaken: empirical software engineering, controlled experiments, interviews, and data analysis. He verifies sources meticulously, curated data, and constructed three data

sets to share with other researchers.

Mr. Arthur Marques showed in his Dissertation that, given the presented context and scope of this research, semantic-based techniques perform equivalently well across multiple artifact types when compared to the state of the art, and that a tool that automates the provision of task-relevant text assists developers in effectively completing a software development task.

Finally, he conducted the research and was the main author of papers that presented associated research findings to the broader SE research community. The publication of full papers in high-quality conferences, partially attested that his work brings significant and original contribution to SE knowledge. Table 1 summarizes the scholarly merits of the candidate, according to the *Instructions for Preparing the External Examiner's Report* provided by the University of British Columbia.

| | Criteria | |
|---|---|---|
| (i) | presents a contribution to knowledge | YES |
| (ii) | is likely to have an impact on the discipline and/or in an applied domain | YES |
| (iii) | describes a coherent body of work whose depth and scope justify the granting of a doctoral degree | YES |
| (iv) | The research undertaken is contextualized clearly, and accurately references the larger field of knowledge on the topic | YES |
| (v) | The methods used are suitably described, relevant to the research question(s), and employed appropriately. | YES |
| (vi) | The research results are reported fully and clearly. | YES |
| (vii) | The analyses and conclusions drawn from the research are well-justified and integrated into the larger field of knowledge | YES |
| (viii) | The implications and limitations of the research are fully discussed | YES |
| (ix) | The writing of the document is of a professional standard | YES |

Table 1: Scholarly Merits

**RECOMMENDATION:** In my evaluation, the Dissertation by Arthur Marques fulfills all requirements for obtaining the PhD degree as stated previously and, therefore, I recommend the candidate and his Dissertation for

defence.

# 4 Questions for Oral Defence

Studies with human based on subjective perceptions and possible concurrent bias from more experienced developers require a lot of effort to be performed and their results analyzed and discussed. I congratulate the author for embracing such line of investigation and bringing interesting new knowledge and technology to assist developers in their daily tasks, based on Android corpus, Python tasks and participants (developers) feedback.

1. *A developer can effectively complete a software development task when automatically provided with text relevant to their task extracted from pertinent natural language artifacts by a generalizable technique.* What do *effectively* and *pertinent* mean in this context?

2. Do you think that the recommended practices adopted by a software development team can affect what a *pertinent* artifact is and selected techniques? Could you elaborate on that?

3. Besides development tasks, can your approach be used in other software engineering tasks?

4. How can your approach relate to well-known software requirements practices and (natural language) artifacts? Could the developer be interested in using them? In which scenarios?

5. Generalizable techniques depends on the existence of similar patterns across different types of artifacts. What if no such patterns are found? Could you state it in a domain-independent way (Android applications, etc.)?

**Salvador, July 11, 2022**

**Dr. Christina von Flach Garcia Chavez**