Towards Predicting Maintainability for Relational Database-Driven Software Applications: Extended Evidence from Software Practitioners

Mehwish Riaz, Emilia Mendes, Ewan Tempero

Department of Computer Science, The University of Auckland, New Zealand

1 mria007@aucklanduni.ac.nz, 2 emilia, 3 e.tempero@auckland.ac.nz

Abstract

The accurate maintainability prediction of relational database-driven software applications can improve the management of projects relating to these applications, thus benefitting software organisations. This paper gives an up-to-date account of the state of practice in maintainability prediction for relational database-driven software applications and provides a baseline for conducting further research in this area. The research involved conducting twelve semi-structured interviews with software professionals, which were then analysed using content analysis with the help of NVivo. The results provide both an account of the current state of practice in that area and also a list of potential maintainability predictors for relational database-driven software applications. These predictors relate to database schema, front-end application, and the interaction of database schema with the front-end application. These results provide the basis for further work involving the proposal and empirical validation of maintainability prediction models for relational database-driven software applications.

Keywords: Software Maintainability, Content Analysis, Interviews, Relational Database-Driven Software Applications, Prediction, Factors.

1. Introduction

Software maintainability [14] is a key quality attribute of software systems. It is defined by IEEE as “the ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment” [14]. Intrinsically associated with it is the process of software maintenance [27], which has long been known to consume the majority of the cost of a Software Development Life Cycle (SDLC) [2, 18]. Software maintainability can significantly impact software costs [27]; therefore it is imperative to understand software maintainability in order to control and improve it.

In order to improve software maintainability, it is important to understand what factors impact upon it, how they can be measured, and how they can be used to predict software maintainability. With the help of a software maintainability prediction model, organizations can make informed decisions about managing their maintenance resources and can adopt a defensive design [25].

Database-driven applications have gained substantial importance in the modern software development [10] and anecdotal evidence suggests that most commonly used databases are relational [7]. Database-driven applications consist of a database, a Database Management System (DBMS), and a set of applications that interact with the database through this management system [15]. When software requirements change, these applications undergo maintenance potentially causing changes to both the software application and the database [10]. Such changes often result in storing an increased number of data sources and relationships [22], which leads to an increase in the complexity of the database schema and in the coupling between the database and the software application [22]. Hence, the maintainability of relational database-driven applications is not only impacted by application-specific features but also by database-specific features.

Due to the importance that relational database-driven applications have within the scope of modern software development, it is imperative to be able to measure and forecast their maintainability. There is, however, little evidence in the literature relating to the maintainability of these types of applications [10, 11], hence motivating the need to gather further evidence on this area.

The aim of this research is to use evidence from industrial practice in order to help improve the field of software maintainability prediction for relational database-driven software applications by making a contribution to the body of knowledge. The evidence has been gathered from 12 interviews with software practitioners (six from Pakistani organizations and six from New Zealand organizations). This paper extends the work presented in Riaz et al. [29] by:

- Provide further details and insights into the evidence gathered on the state of practice regarding maintainability prediction for relational database-driven software applications. Specifically, it supplies further evidence in the form of quotes from the evidence stated by the interviewed practitioners.
- Present an extended list of factors to be taken into consideration for understanding and predicting the maintainability of relational database-driven software applications. Note that Riaz et al. [29] provided the list of factors mentioned by only 3 or more interviews. This paper lists factors presented by 2 or more interviewees.
- Discusses in-depth the details of the interview, data collection, and data analysis procedures. Specifically, it discusses the procedure used for coding the conducted interviews using NVivo which may be used as a guideline for coding interviews conducted as a part of content analysis in the area of Software Engineering (SE).

The remaining of this paper is organised as follows. Section 2 presents the background and related work for this research, followed by the description of the research methodology in Section 3. Section 4 presents the results from the interviews; a discussion on the results and threats to validity are given in Section 5 followed by conclusions and future work in Section 6.

2. Background

The research described herein was informed by the results of a Systematic Review (SR) on the topic ‘software maintainability prediction and metrics’ [27]. The results of the SR revealed that there was very little evidence on software maintainability prediction. Specifically, existing maintainability prediction models were built following algorithmic approaches, in particular the regression-based models. In addition, these models were specific to the datasets used to build them, and therefore their external validity was low; the common predictors used by these models were those related to application size, application complexity and coupling; and the most common measure for maintainability was an expert judgment.
made on maintainability represented as ordinal scale measure. The 15 studies selected during the process of SR [27] were further analyzed to assess if the datasets used in these studies completely or partially comprised relational database-driven software applications. Only three of these studies [8, 16, 17] had used relational database-driven software applications for the purpose of their research evaluation. However, only one of these studies [16] presented a maintainability prediction model but did not provide its prediction accuracy; the other two studies only presented measures [8] and factors [17] that impacted software maintainability, without any associated maintainability prediction model. Further analysis of these studies revealed that although they used relational database-driven applications, none of their proposed predictors or factors related specifically to a back-end database or to the interaction between a back-end database and the front-end application.

In addition to the SR above mentioned, a complementary literature review was carried out focusing on the topic of relational databases. This review showed that while there was evidence on numerous topics relating to relational databases (schema evolution [5, 9], schema versioning [9, 13], testing of database-driven applications [3, 6, 11, 15], maintenance of database applications [10], impact analysis and prediction of database schema changes [22], evaluation of maintainability of persistency techniques [12]), there was no evidence on maintainability metrics or prediction for relational databases or relational database schema.

The lack of existing literature focusing specifically on maintainability prediction for relational database-driven software applications prompted us to carry out an investigation with software practitioners in order to gather data on relational database-driven applications maintainability predictors and metrics used in practice.

3. Research Methodology

The research methodology used herein was mainly qualitative in nature. A total of twelve semi-structured interviews with thirteen software professionals were conducted, which were analyzed with the help of content analysis [20].

3.1. Semi-Structured Interviews

Interviews – a commonly used technique for collecting qualitative data – can be used to collect opinions or impressions about a phenomenon of interest and to help identify the terminology used in a given setting [19]. For this research, semi-structured interviews were conducted. This type of interviews were best suited for this research as these comprise a mix of open-ended and closed-ended questions and aim at not only eliciting expected information, but also unforeseen information [19].

The interviews were planned such that they were conducted in two steps. In the first step, six interviews were carried out with software professionals in Pakistan. In the second step, which was planned so that it would be informed by the responses obtained from the first step, six interviews were conducted with software professionals in New Zealand.

Each of the interviews began with a short introduction to the research, where it was emphasized that the research focused at relational database-driven software applications; the research objectives were also briefly explained prior to asking the main questions. The questions asked during the interviews are given in the appendix. A pilot run of the interview gave an estimated time ranging from 20 to 45 minutes depending on whether the responses were straightforward or very detailed. In the first phase of the interviews which involved software professionals from Pakistan, information from the SR was shared with the interviewees, only as an example, to elaborate the concept of maintainability prediction and to
enable better information elicitation. This was done using simple language. Similarly, in the second phase that involved interviewing software professionals in New Zealand, information from the first phase was shared with the interviewees as we believed this would give interviewees a better idea about the sort of information in which we were interested. This also helped keep both the interviewer and interviewees focused.

3.2. Data Source

The data source used herein comprised interviews with 13 software professionals from 10 different software companies in Pakistan and New Zealand. A total of 14 companies were initially contacted, of which 10 agreed to participate within the allocated time-frame for data collection. A total of 12 interviews were conducted, six in Pakistan and another six in New Zealand. Note that in one of the interviews conducted in Pakistan, two interviewees participated; however, the information provided by them overlapped; therefore, in the subsequent Sections of this paper their responses are considered to come from a single source (i.e., single interviewee). The interviewees were either Project Managers (PM), project Team Leads (TL) or Quality Assurance (QA) professionals. These roles were chosen motivated by the fact that software companies differ in their size and the types of projects they manage; therefore, for a given role the associated responsibilities defined in different companies are likely to differ. We have observed that in some large companies a Software Engineer (SE) is only responsible for coding, whereas in others a SE participates in all phases of a software development life cycle (SDLC). Therefore, there was no certainty as to whether developers or maintainers were involved in other phases of a SDLC in addition to implementation. For the purpose of this research, the best suited roles were those that were involved in all phases of a SDLC and had a considerable level of experience dealing with quality-related issues and maintenance tasks. PMs, TLs and QA professionals best fitted these requirements as they were better able to reflect upon their experiences and lessons learned from the projects they had been involved in.

The two countries, Pakistan and New Zealand, were chosen for the following reasons: i) the first author is originally from Pakistan; ii) most software companies in Pakistan deal with outsourced projects from the United States, United Kingdom, Denmark, etc. These companies are well established and follow commercial quality standards and practices. We believed that we could gather rich data from these companies; iii) the research is being carried out at the University of Auckland, New Zealand to which all the three authors are affiliated; and iv) we believe that the context of this research was not culture-sensitive as the focus was on software applications in general and not specific aspects related to people or their way of carrying out their work.

The interviewees were all at least at managerial positions in well established companies; one CEO and founder of a company was also interviewed. The interviewees had an average experience of 10.3 years in software development and management. A summary of interviewees’ experience and countries to which each role belonged is given in table 1. All the companies where the interviewees worked, except for one, developed only relational database-driven software applications. This was also confirmed during the interviews.
### Table 1. Interviewees’ Summary per Role

<table>
<thead>
<tr>
<th>Role</th>
<th>No. of Interviewees in Pakistan</th>
<th>No. of Interviewees in New Zealand</th>
<th>Total Years of Experience (Averaged per role)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM</td>
<td>6</td>
<td>4</td>
<td>10.35</td>
</tr>
<tr>
<td>QA</td>
<td>1</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>TL</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>CEO &amp; Founder</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
</tbody>
</table>

#### 3.3. Interview Procedure

The research questions from the SR [27] on software maintainability prediction were used as a basis to develop the questionnaire used herein. In order to conduct these interviews, approval from University of Auckland Human Participants Ethics Committee (UAHPEC) was obtained - ref. 2009/527.

Contacts were made with the companies and the participating employees both through email and telephone. The interviews in Pakistan were conducted in July 2009 and those in New Zealand were conducted between August and November 2009. All the interviews, except for one conducted in a coffee shop, were carried out by the first author at the participating companies’ premises. All the interviews were face-to-face and recorded on two digital recorders (the second recorder was used as a backup). During the interviews, extensive notes were also taken. This was done to record important terms and points, and also as means to detail further certain points without abruptly cutting the interviewee off and breaking their chain of thought.

The format of the questions and the order in which they were asked during each interview was the same. In the first step of interviews, conducted in Pakistan, the language used during the interviews was Urdu – the native language of the country. This was used so the interviewees could fully express their thoughts and perceptions. In this step, the interviewer shared her perception of maintainability with the interviewees, after asking them about their own views on the topic. This was done so that both the interviewer and interviewee were on equal grounds to discuss further concepts. The interviewer also shared some predictors of maintainability gathered from the SR conducted on software maintainability prediction and metrics with the interviewees to give them an example of the perception of predictors.

The second phase of interviews was conducted in New Zealand in English. The same process was followed, except that some predictors specific to database-driven applications gathered from the first phase of interviews were also discussed with the NZ interviewees. In both phases, the duration of the interviews ranged from 20 to 60 minutes with an average duration of 38 minutes.

#### 3.4. Data Analysis

Prior to analyzing the data obtained from the interviews, all the conducted interviews had to be transcribed. The six interviews conducted in Pakistan had also to be translated from Urdu to English. Translation and transcription were each very cumbersome tasks and their combination even more so. Each recorded hour in Urdu took on average 10 hours to translate and transcribe and each recorded hour in English took on average 8 hours to transcribe. All the interviews were translated and transcribed by the first author due to her familiarity with both languages, and also to reduce the chances of missing any information given that the same person conducted the interviews, and therefore could also easily refer to the notes taken.
during the interviews. The translations and transcriptions were further verified by a volunteer who listened to parts of interviews at random and tallied them with the time stamped transcripts of the interviews, thereby validating whether what he heard was consistent with what has been transcribed.

The general steps followed for the qualitative analysis are given below [4]:
1. Organize and prepare data for analysis
2. Read through all the data
3. Begin detailed analysis with a coding process
4. Generate categories or themes for analysis
5. Interrelate categories or themes
6. Interpret the meaning of categories or themes

The specific technique used for qualitative data analysis was ‘content analysis’, using the framework and components defined by Krippendorff [20]. Content analysis is defined as “a research technique for making replicable and valid inferences from texts (or other meaningful matter) to the contexts of their use” [20]. It was chosen over other qualitative data analysis techniques because of its unobtrusive nature, context sensitivity, and ability to handle large amounts of data [20]. The components that were used to proceed from text to results are as below:

- **Unitizing** – the systematic distinguishing of segments of text that are of interest [20]. This was done in accordance to the information intended to gather i.e., the text was segmented according to the interview questions which was then used for further coding.
- **Sampling** – allows the analyst to economize on research efforts by limiting observations to a manageable subset of units that is conceptually representative of the population [20]. This was carefully done by choosing appropriate companies and roles within the companies to have a better representation of the population.
- **Recording/Coding** – transformation of unedited text or unstructured sounds into analyzable representations [20]. This was also carefully performed – recording done on reliable digital recorders and coding done with the help of NVivo [24] against the units of analysis such that it reflected the research goals and was tied up to particular research questions [21]. The transcribed interviews were stored in NVivo Sources as Internals. The questions asked during the interviews were used as the Tree Nodes and the variety of answers as child nodes. The process followed for coding was that the text was read from the Internals sentence by sentence. If a sentence or a part of it corresponded to a unit of analysis that was of interest, it was selected and a node was chosen for it, if the node already existed. If a suitable node did not exist, it was created as a child node of an existing node e.g., in cases where a different answer to a question was found or a new factor was found in an interview not reported in previously coded interviews. The coding also enabled the results to be summarized for number of Sources.
- **Reducing** – serves the need for efficient representations [20]. This was done alongside coding.
- **Inferring** - bridges the gap between descriptive accounts of texts and what they mean [20]. This was done against the questions asked during the interviews which were essentially open-ended versions of the research questions defined in the SR but specific to database-driven software applications.
- **Narrating** - amounts to the researchers making their results comprehensible to others [20]. This has been done in following Section.
4. Results Informed by Industrial Practice

The results obtained by performing content analysis on the conducted interviews, are given below. The subsections have been organized according to the questions asked during the semi-structured interviews. Also, since the focus of this research was made clear to all interviewees, prior to asking them any questions, it was understood by the interviewees that the context of the interview questions was specific to relational database-driven software applications.

4.1. Perception of Maintainability

The most common perception of maintainability was to consider it as the ease or difficulty to implement changes in the software or to carry out maintenance tasks. This was reflected in 7 of the 12 interviews (58%). Four of the interviewees perceived maintainability as the time taken to fix a software bug. For them software is maintainable “if it takes less time to fix bugs in it”. The term ‘ease’ to them was a “bit of a relevant term”, and was understood in the context of time. Two of the interviewees also mentioned “availability or low down time” in the context of database-driven applications stating “the major aspect of maintainability in case of databases is that it should be available 24/7”.

4.2. Should Applications be Maintainable?

All the interviewees agreed that software applications should be maintainable. The reasons provided included “scalability” - to be able to incorporate future changes; producing “quality product” and “mature products which are more reliable”; making an application “customizable” because “you do not know what feature a client or a client of client may require in future”; reduced “resource utilization”, “cost”, “effort” and “development time for adding new features”; “convenience to the developers and customers”; and “client satisfaction”.

4.3. Practical Experience with Maintainability Prediction

The analysis of the interviews revealed that software maintainability prediction is not a common concept in practice. This was reflected by the responses of all, except two, interviewees, who in answer to whether they have any experience with maintainability prediction said “yes in order to improve” and “yes we can see that it (software application) will not last forever and…due to the older technologies that we adopt (software application) becomes obsolete and then we have to move forward”. None of the interviewees had experience with predicting software maintainability whereby it is done by following any specific prediction model or formally done as part of the development or project planning processes. In particular, two of the interviewees stated, “I have not seen such scenarios as a practitioner” and “we aim to create maintainable software but no we do not do any such thing explicitly”. However, all the interviewees agreed that maintainability should be predicted and also stated the perceived benefits associated with its prediction, see Section 4.4.

4.4. Benefits of Predicting Maintainability

The benefits identified by the interviewees to predict software maintainability were: i) improved costs and effort, given that “we cannot spend cost and effort on the same thing over
and over again”; ii) to help organizations in “decision making” and “resource planning” in the context of “where we should put our money and time”; and iii) improved design and performance in terms of “we can change the design to make it easier to maintain” and “it (maintainability prediction) can show what parts of the product will require changes in the future and so we can create better documentation to support future modifications”. Note that the interviewees used the terms design and software architecture interchangeably.

Other benefits narrated by the interviewees included “deep understanding of the application”; ability to “negotiate with client” on cost, and allowing the team to spend “more time on quality” and reducing “time spent on operations”; and the ability to decide whether to bid on a project by considering “not just the initial cost of project’s development but life time cost”.

4.5. Is Maintainability Predicted in Practice?

The responses of the interviewees, when asked whether maintainability was predicted in their respective companies, were mixed (see Table 2). Eight interviewees (67%) gave a positive response to this question. They said that maintainability is predicted based on “expert judgment” which “depends on previous experience”. They “can determine the maintainability as high or low” by “looking at the application”. One interviewee gave a negative response. Since his company developed and managed applications that interacted with large databases, his context was that of database-driven applications and the features provided by the tools used to develop these applications. He also stated that “the features are not available. Those that are available are practically not implementable and have a high overhead”. Two more interviewees did not give a definite answer. They initially answered that maintainability was not predicted in their company; however later added that they make a “judgment that is very subjective” and that it is not predicted “generally speaking from some measure but there are certain things that I do know”. None of the interviewees suggested any quantitative measures to measure maintainability.

Table 2. Do Software Companies Predict Maintainability?

<table>
<thead>
<tr>
<th>Is maintainability predicted in the company?</th>
<th>No. of Interviewees (out of 12)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>8</td>
</tr>
<tr>
<td>No</td>
<td>2</td>
</tr>
<tr>
<td>50 - 50</td>
<td>2</td>
</tr>
</tbody>
</table>

In relation to at what project stage maintainability can be best predicted, seven interviewees (58%) supported the design stage. According to their point of view, ‘design’ or ‘software architecture’ was the best project stage to predict software maintainability. They said that they could make a better judgment based on design than a judgment made by considering resources, deliverables, and even the source code. The remaining interviewees gave no definite answer to this question. The supporters of design chose it because it can be improved before the implementation: “it can show the visibility to make changes or to enhance functionality” and “application is the last thing to be challenged”. One of the interviewees also mentioned that they have a formal review meeting on the design and the design is not forwarded for implementation unless “we have improved it”. This means that at least those we interviewed believed that the maintainability of a relational database-driven software application can be subjectively predicted made based on an application’s design or software architecture.
4.6. Maintainability Measures

All the interviewees, except for one, believed software maintainability was nearly impossible to measure. They stated that “we cannot say it on the basis of any measurement that how much maintainable it (software application) is or it is not”. However, they were of the view that software maintainability can be judged by considering various factors such as those mentioned in Section 4.8. The only interviewee who believed maintainability could be measured supported the idea of “measuring maintainability in terms of man hour resources” or “in terms of dollars” as he believed that “ultimately (it) comes down to it”.

4.7. Is There a Difference between Maintainability for Relational Database-Driven Applications and Applications that do not have any Back-End Database?

All the interviewed practitioners, except for one, corroborated that the maintainability of relational database-driven software applications is different from that of software applications that do not have any back-end database by stating “if we consider the database-driven applications, not only the factors related to database play a part but also those factors that are because of the technical aspects of the DBMS are introduced”. One of the interviewees believed it was the case “in terms of scalability (and) performance”. Another interviewee, who was a very strong supporter of database-driven applications, stated that “I call those applications without a database more as tools, they are not applications. To me applications run on the database”. However, one of the interviewees also believed that although the difference exists “the boundary between database-driven and non-database thing is fading as more object oriented stuff is going into object level persistency”. The interviewee who believed there were no differences in the maintainability of database-driven and non database-driven software applications was of the view that “if it is a database-driven application, then one of your modules is database” and that implementing changes to a database schema should be considered the same as implementing changes to different parts of an application.

4.8. Factors to be Considered for Predicting the Maintainability of Relational Database-Driven Software Applications

Several factors specific to predicting the maintainability of relational database-driven software applications were identified by interviewees (see Table 3). Note that Table 3 lists only the factors stated by two or more interviewees. We argue that it is very likely that these factors would also be applicable to other software companies beyond the ones who participated in this research, given that most of the factors shown in Table 3 were selected by companies in both Pakistan and New Zealand, thus suggesting that they are applicable across countries.

<table>
<thead>
<tr>
<th>Factors</th>
<th>#Interviewees Pakistan</th>
<th>#Interviewees New Zealand</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application’s architecture or design</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Tools &amp; technologies including development frameworks</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Database (DB) design including the following:</td>
<td>5</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>1. proper normalization</td>
<td>5</td>
<td>2</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 3. Factors for Predicting Software Maintainability Specific to Relational Database-driven Software Applications
The factors listed in Table 3 suggest that the most important part in relational database-driven software application’s maintainability is played by the application’s architecture,
followed by the database design. This suggests that it is most important for organisations to
design their applications correctly. In addition, other factors also seem to be relevant
predictors of maintainability, such as factors related to code and design, and also related to
software processes and people.

Note that the factors listed in Table 3 may seem to be at different levels of abstraction
and/or related; however we refrained from merging any factors in order to present exactly the
factors as emphasized by the interviewees.

5. Discussion

This paper presents evidence from the state of practice on maintainability prediction for
relational database-driven software applications. The results suggest that maintainability is
understood in practice as a quality attribute. However, maintainability is also sometimes
understood in terms of time taken to correct issues in the software application. This can be
attributed to the fact that quality itself is rather an intangible concept and so maintainability as
a quality attribute is dealt with as a relative concept understood in terms of time.

It also appears that a formal prediction model, or approach to predicting the
maintainability of relational database-driven applications is not used in practice. However, the
interviewees did suggest that they relied upon expert opinion when making a judgment on the
maintainability of such applications. Therefore the notion that maintainability is unlikely to
be predicted in practice may be slightly misleading as ‘expert opinion’ also is a well-known
prediction technique [23]. Nevertheless, the use of subjective means for prediction is known
to present several problems [23]; therefore there is a need to formalize this process by
proposing and validating maintainability prediction models that use maintainability predictors
that are relevant in practice. This will enable their adoption by the practitioners.

It should also be noted that while the SR research questions were used as a basis to
prepare the questions used in the interviews, these questions differed from those of the SR.
The questions used in the interviews were mainly open-ended and there was a provision for
detailed discussion such as by using ‘why’, ‘how’ and ‘what’ probes. Therefore, for such
cases where the interviewees mentioned use of any software maintainability prediction model,
they could be interrogated further on the details specific to the models, prediction accuracies,
cross-validation methods, etc. that they used for predicting maintainability of relational
database-driven software applications.

Based on the evidence presented in this paper, we believe that it can be argued that the
maintainability of relational database-driven software applications is perceived to be different
from that of the applications that do not interact with a back-end database. In this regard, the
interviewees suggested the factors they believed to be important to consider when assessing
the maintainability of such applications. Also, a survey designed to further validate the
predictors presented herein and to determine the level of their importance to the practitioners
is under progress.

We believe it is important to stress that the sampling for this research, as also discussed in
section 3.2, was carefully done. The companies chosen were such that they represented small
as well as large organizations; dealt with small-scale as well as large-scale applications; did
in-house development as well as outsourced projects; and were based in one location as well
as had operations in more than one city or country. Therefore, we believe that the sample used
is representative of software companies that develop relational database-driven software
applications. Further, an analysis of the evidence from the industry interviews showed that the
answers given by the Pakistani and New Zealand companies are very similar; these results
imply that maintainability prediction for relational database-driven software applications may
not be sensitive to culture, and in addition, results may also be generalized outside the sample population used in this research.

As with most research, this research work also has limitations and threats to validity. A first limitation is that the conducted interviews are based on the recollection of interviewees’ experiences and hence present a possibility of missing details. This limitation was addressed for most part by using appropriate prompts during the interviews to probe further on important details.

A second limitation can be that the interviewees during the first phase of the interviews were also told some of the results from the SR conducted on software maintainability prediction and metrics, to use as an example, and the interviewees in the second phase were informed of some of the factors identified in the first phase of interviews. While this had an associated disadvantage of introducing the interviewer’s own perceptions and threatening the validity of the data [1], after much contemplation this was considered important especially in cases where it was absolutely necessary to enable the discussion with the interviewees to take place. In addition, the interviewer tried to only relay the findings in a considerably neutral tone in an attempt not to influence interviewees’ thoughts on the topic. Also, upon further analyzing the data obtained from the interviews, it was observed that while some of the responses validated the evidence gathered in the previous phase, additional evidence was also gathered. Hence, in each step, information was added up.

A third limitation can be due to inaccurate hearing of the recorded interviews. This was minimized by spending a generous amount of time on translation and transcription, extensive pause-and-play to hear unclear words and phrases, and by having a volunteer validate a subset of the transcripts.

A fourth limitation relates to the external validity of the results. This is because the results come only from twelve interviews, which can be argued to be a small sample; however the nature of the study is qualitative and does not involve statistical analysis where large volumes of data is a requirement for ensuring a certain degree of external validity. Also, the sampling, as discussed above, was also done carefully to ensure that the data was representative of the large population. In addition, we also had to take into account constraints such as the time available to carry out this work, and also the amount of time each interview took (to be carried out and later translated and transcribed).

6. Conclusions

This paper presents the evidence on software maintainability prediction for relational database-driven software applications. The research involved conducting twelve interviews with software professionals. The paper discusses the perception of software maintainability; the perceived benefits of predicting it; its measures; the techniques and models used for its prediction; and its predictors or the factors that impact upon it and are specific to relational database-driven software applications - all from evidence gathered from state of practice.

The major contribution of this paper is that it presents the current state of practice on software maintainability prediction for relational database-driven software applications. The results suggest that maintainability for relational database-driven applications is understood in practice as a quality attribute; it is not measured or predicted using a formal metric or prediction technique but the practitioners believe that it should be predicted. The practitioners also believe that the maintainability of relational database-driven software applications is different from that of the applications that do not have a database back-end. The results from the interviews also suggest that maintainability prediction in the domain of relational database-driven applications is done based on expert judgment. This relates to the fact that
expert judgment is made on maintainability because it is required for these applications and because there is no formal prediction model for quantifying these judgments. This suggests a strong need for a formal technique that can be used to quantify maintainability prediction. The predictors provided in this paper take formalizing the process of maintainability prediction for the mentioned type of software applications one step forward. These factors, even if taken into consideration while developing or maintaining these applications, can result in improved software maintainability, as the collected evidence suggests.

The future work involves further verification of these factors with the help of a survey, followed by the use of these factors in software maintainability prediction models built from real project data gathered from software companies, and the assessment of these models’ accuracy.

References


Appendix: Interview Questions

1. What is your understanding of maintainability?
2. Do you believe that software applications should be maintainable? If yes, why?
3. What is your view and/or experience with software maintainability prediction?
4. What, in your opinion, may be possible benefits of predicting software maintainability?
5. Is maintainability measured and/or predicted in your company, even if it is based on expert judgment?
   a. If yes,
      i. What are the factors that you take into account when predicting maintainability?
      ii. How do you measure maintainability?
   b. If no,
      i. Why not?
      ii. What would you consider in terms of artefacts e.g., design, resources, deliverables, source code etc. in order to predict maintainability?
      iii. What factors would you consider in order to predict maintainability?
6. When dealing with maintainability, do you differentiate between database-driven and non database-driven applications? Why? What are the additional factor(s) that need to be considered when developing a database-driven software application?

Authors

Mehwish Riaz is a PhD Student at the Department of Computer Science, The University of Auckland, New Zealand. She also holds a Masters in Science in Software Engineering degree. She has published her research at various reputed International conferences. Her research interests include empirical software engineering, software maintainability, software quality prediction, software architecture and systematic literature reviews.
**Emilia Mendes** is an associate professor of computer science at the University of Auckland, New Zealand. She has active research interests in the areas of empirical Web and software engineering, evidence-based research, hypermedia, and computer science and software engineering education, in which areas she has published widely, with more than 115 refereed publications, which include two books (one edited (2005) and one authored (2007)). She is on the editorial board of the International Journal of Web Engineering and Technology, the Journal of Web Engineering, the Journal of Software Measurement, the International Journal of Software Engineering and Its Applications, the Empirical Software Engineering Journal, and Advances in Software Engineering.

**Dr. Ewan Tempero** is an Associate Professor in the Department of Computer Science at The University of Auckland, New Zealand. He graduated from the University of Otago, New Zealand, with a BSc (Honours) in Mathematics in 1983 and received his PhD in computer science from the University of Washington, USA, in 1990. He has published over 100 papers in journals and internationally-refereed conferences, mainly in the areas of software reuse, software tools, and software metrics. His current research is developing metrics for measuring the quality of software designs.