A Tool-based Boilerplate Technique to Improve SRS Quality: An Evaluation

S.Ahmad, U.Anuar, and N.A.Emran
Faculty of Information and Communication Technology
Universiti Teknikal Malaysia Melaka
sabrinaahmad@utem.edu.my

***Abstract*— This paper presents an evaluation of a boilerplate technique with the assistance of a tool-based prototype in order to improve Software Requirements Specification (SRS) quality in terms of comprehensibility, correctness and consistency. The value behind this boilerplate is to ease the process of identifying essential requirements for a generic information management system and translating them into standard requirements statements in the SRS. An empirical investigation environment is adapted and expert judgment method is used for evaluation. Results showed that the tool-based boilerplate technique improves the completeness, correctness and consistency of requirements in SRS.**

***Index Terms*— Empirical Investigation; Expert Judgment; Software Requirements; SRS Quality.**

I. Introduction

A boilerplate is usually used in industry to produce consistent and symmetry goods. The same understanding is adapted to the software engineering specialization to particularly produce an SRS document to meet some standard. The boilerplate technique ability to produce requirements statements in some sort of controlled environment is seen as beneficial to reduce the possibility of defects. The control is made to handle the flexibility of natural language (NL) which usually leads to ambiguity and inconsistency problems.

The tool-based boilerplate is a prototype developed for the usage of producing an SRS for an information management system. Basic functions of an information management system are mainly the same and can simply adapt to a variety of systems. This prototype only covers Section 3.2 of SRS which describes functional requirements. The layout is based on IEEE template [1]. The philosophy behind this boilerplate is to provide a guide to properly write requirements statements to improve comprehensibility. In addition, an auto generated wireframe interfaces will improve the correctness of the requirements stated. Finally, the uniform sentence structure build-in the boilerplate will improve the consistency and reduce the possibility of having ambiguity. There are three basic guides to write proper requirements statement as stated in Table 1.

The objective of this paper is to show that the boilerplate technique is proven beneficial to improve the quality of SRS. Following the introduction, section 2 elaborates on background work on boilerplates for software engineering. This is followed by Section 3 which explains the proposed tool-based boilerplate technique. Next, Chapter 4 presents the evaluation based on expert judgment and Section 5 concludes the paper.

Table 1
Basic Guides for Writing Good Requirements Statements

Verbs	Purpose
SHALL	Is used to indicate a mandatory requirement. It must be implemented and verified.
SHOULD	Is a non-mandatory requirement, used to indicate a goal which must be addressed but it not formally verified.
WILL	Is used to indicate a statement or fact which is not subject to verification.

II. Background Study

The requirement is a basis of a system that molds the shape of the system to be developed in a software project. SRS is a fundamental document which consists of a set of requirements that forms the foundation of software development process. Since the importance is obvious, the quality of the SRS is crucially important. Poor quality SRS does not only lead to increase in development and sustainment costs but also cause major schedule overruns [2].

We reviewed literature ranging from the year 2000 to 2014 on boilerplate’s contribution to improving SRS quality. According to the literature, the boilerplate technique is proposed as a bridge between informal, NL specification and formal specification and was adopted as one of the semiformal languages for the requirement specification [3]. According to a research, boilerplate for requirements specification documentation consists of a set of predefined templates and there are 3 main categories of boilerplate (capability, functional and constrains) [4]. The boilerplate is used in tools to semi-automatically transform NL requirements into semi-formal boilerplate requirements, called DODT. The tool is believed can reduce the manual effort of the transformation and to improve the quality of the requirements. Besides, EARS boilerplate is one of the well-known boilerplate that have been using to overcome the NL problem. EARS provides five different requirement types translated into EARS template and reduce the ambiguity, vagueness and wordiness of the requirements [5]. Furthermore, boilerplate is effective to reduce ambiguity and making them more amendable to automated analysis. It provides a simple and yet effective way to increase the quality of requirements by avoiding complex structure and inconsistency in requirements [6]. In addition, boilerplate is also used for several other purposes; to enable the reuse of parts of the requirements text, to increase the quality of requirements and to ensure completeness of the specification. This is also supported by research stating that, the predefined boilerplates were found helpful in collecting consistent and standardized requirements, especially among novice requirements engineers [7]. Besides, it can be used as a preliminary basis for requirements checking. Digging further on literature, boilerplates also can be used for several classes [8] of requirements such as capability, function, timeliness, mode and operational constraint [9]. Research has been done to discover the coverage of boilerplate's research on improving quality attributes of SRS. According to [10], unambiguity is a quality attribute where boilerplate contributes the most (30%). This is followed by conformity (22%), completeness, accuracy and reusability (13%) and other quality attributes (9%). It is suggested that the quality problem in SRS is multidimensional and boilerplate technique has been found useful to improve SRS to overcome several quality problems [11].

Being a requirements engineer, it is always an obligation to get a complete set of requirements [12]. The completeness is important because of the (1) acceptance and satisfaction, (2) development cost and schedule, (3) development, (4) verification and (5) safety. More importantly, complete requirements leave no room for readers to misinterpret the requirement. This is crucial because readers tend to make an assumption and fill in the missing information if the requirement is incomplete. Besides, different readers with different background will interpret the requirements differently based on their own experience and understanding. Therefore, it is equally important to get a comprehensible set of complete requirements. Comprehensibility is achieved if all requirements are specified and phrased in a way that is easily understood [13]. This is sometimes referred to as 'clear' or as 'understandable'.

Correctness can at least be considered from two different perspectives which are (1) from a normal point of view, where usually correctness is a combination of consistency and completeness and (2) from a practical point of view, where correctness is defined as a certain business goal [13]. This research is considering the second perspective as the boilerplate provides a guide to have essential requirements.

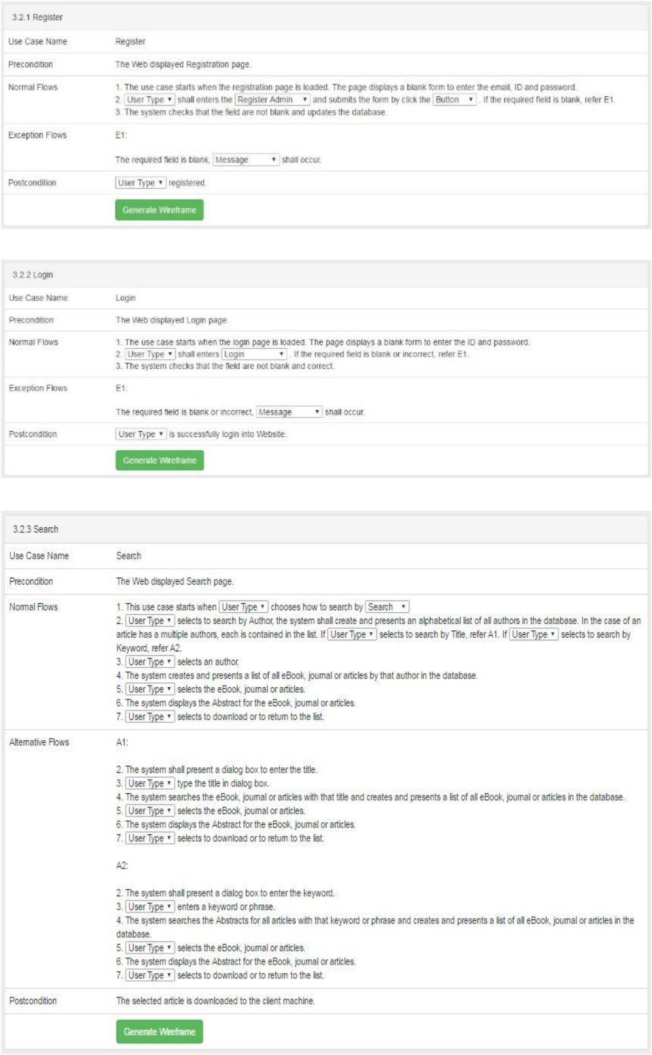
Consistency is referring to a situation where a requirement contains no internal contradictions [14]. It is usually divided into two which is internal consistency and external consistency. In this paper, we only focus on internal consistency which is included in the SRS.

III. OUR TOOL-BASED BOILERPLATE TECHNIQUE

Our tool-based boilerplate is designed to aid requirements engineer to elicit requirement from stakeholders and to record the requirements statements appropriately in SRS. The aim of this tool is to guide the requirements engineer to write the SRS with consistency. Besides, the boilerplate is also designed to assist the requirement engineers to write complete and correct requirements. The effort will improve the SRS reliability and enhance the comprehensibility of the requirements statements for both developers and customers.

This research scopes the tool prototype to cover Section 3.2 of software requirement specification (SRS) only which covers the elaboration of functional requirements. Besides, the format and the layout used in the tool prototype are based on IEEE template [1]. In this research, the tool-based boilerplate is designed for a generic information system management which usually has similar essential requirements. The essential requirements or also known as basic functionalities cover registration, login, searching, adding information and maintaining information with the edit, delete and update capabilities. These basic functionalities can be utilized by many information system management and further elaborated to make a complete system. Besides, in order to increase the correctness quality of the SRS, a wireframe interface is auto generated to portray the requirements statements stated in the SRS.

Figure 1 shows an example of screen shots of the tool-based boilerplate.



3.2.1 Register

Use Case Name	Register
Precondition	The Web displayed Registration page.
Normal Flows	1. The use case starts when the registration page is loaded. The page displays a blank form to enter the email, ID and password. 2. User Type ▾ shall enters the Register Admin ▾ and submits the form by click the Button ▾ . If the required field is blank, refer E1. 3. The system checks that the field are not blank and updates the database.
Exception Flows	E1: The required field is blank, Message ▾ shall occur.
Postcondition	User Type ▾ registered.
	Generate Wireframe

3.2.2 Login

Use Case Name	Login
Precondition	The Web displayed Login page.
Normal Flows	1. The use case starts when the login page is loaded. The page displays a blank form to enter the ID and password. 2. User Type ▾ shall enters Login ▾ . If the required field is blank or incorrect, refer E1. 3. The system checks that the field are not blank and correct.
Exception Flows	E1: The required field is blank or incorrect, Message ▾ shall occur.
Postcondition	User Type ▾ is successfully login into Website.
	Generate Wireframe

3.2.3 Search

Use Case Name	Search
Precondition	The Web displayed Search page.
Normal Flows	1. This use case starts when User Type ▾ chooses how to search by Search ▾ 2. User Type ▾ selects to search by Author, the system shall create and presents an alphabetical list of all authors in the database. In the case of an article has a multiple authors, each is contained in the list. If User Type ▾ selects to search by Title, refer A1. If User Type ▾ selects to search by Keyword, refer A2. 3. User Type ▾ selects an author. 4. The system creates and presents a list of all eBook, journal or articles by that author in the database. 5. User Type ▾ selects the eBook, journal or articles. 6. The system displays the Abstract for the eBook, journal or articles. 7. User Type ▾ selects to download or to return to the list.
Alternative Flows	A1: 2. The system shall present a dialog box to enter the title. 3. User Type ▾ type the title in dialog box. 4. The system searches the eBook, journal or articles with that title and creates and presents a list of all eBook, journal or articles in the database. 5. User Type ▾ selects the eBook, journal or articles. 6. The system displays the Abstract for the eBook, journal or articles. 7. User Type ▾ selects to download or to return to the list. A2: 2. The system shall present a dialog box to enter the keyword. 3. User Type ▾ enters a keyword or phrase. 4. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all eBook, journal or articles in the database. 5. User Type ▾ selects the eBook, journal or articles. 6. The system displays the Abstract for the eBook, journal or articles. 7. User Type ▾ selects to download or to return to the list.
Postcondition	The selected article is downloaded to the client machine.
	Generate Wireframe

Figure 1: Examples of Screen Shots of the Tool-based Boilerplate Prototype

IV. EVALUATIONS, RESULTS AND ANALYSIS

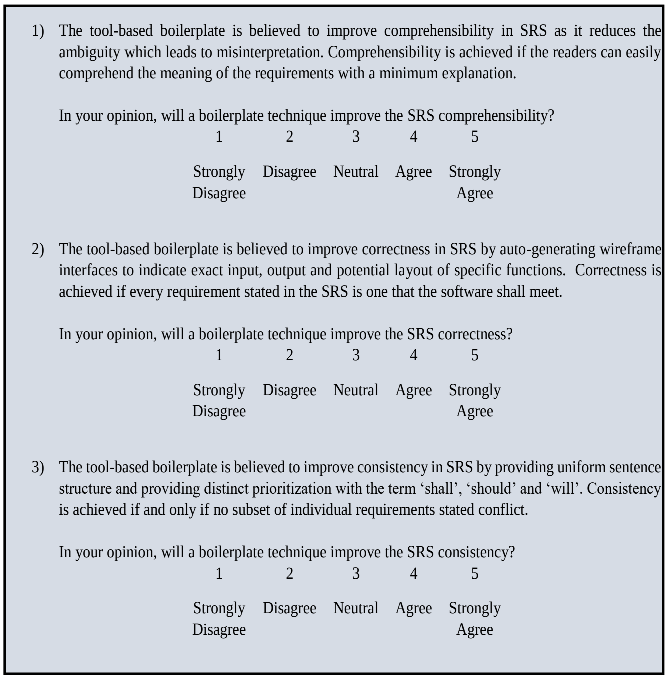
Expert judgment method is used for the evaluation. The underlying principle of the method is the encoding of experts' tacit knowledge into probabilistic measures associated with the achievement level of software requirements quality attributes. An aggregated quality measure is obtained based on experts' preferences related to quality attributes captured through a questionnaire.

A. Identifying Experts

Ten experts were identified to participate in the evaluation. They were practitioners dealing with system development particularly software requirements in Malaysia industry. Their experience ranges from five to twenty years with the different level of seniority. The experts were also formally trained in software engineering good practice with a different level of qualifications namely, first degree, master degree and doctor of philosophy. Several of them also recently joined universities as academics.

B. Instruments

The experts are the instrument for this evaluation protocol as they will provide their expert judgment on the software requirements quality achieved through the tool-based boilerplate technique.



1) The tool-based boilerplate is believed to improve comprehensibility in SRS as it reduces the ambiguity which leads to misinterpretation. Comprehensibility is achieved if the readers can easily comprehend the meaning of the requirements with a minimum explanation.

In your opinion, will a boilerplate technique improve the SRS comprehensibility?

1	2	3	4	5
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

2) The tool-based boilerplate is believed to improve correctness in SRS by auto-generating wireframe interfaces to indicate exact input, output and potential layout of specific functions. Correctness is achieved if every requirement stated in the SRS is one that the software shall meet.

In your opinion, will a boilerplate technique improve the SRS correctness?

1	2	3	4	5
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

3) The tool-based boilerplate is believed to improve consistency in SRS by providing uniform sentence structure and providing distinct prioritization with the term ‘shall’, ‘should’ and ‘will’. Consistency is achieved if and only if no subset of individual requirements stated conflict.

In your opinion, will a boilerplate technique improve the SRS consistency?

1	2	3	4	5
Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree

Figure 2: Questions to gather experts’ opinion

In order to allow the experts evaluation, an empirical investigation environment is adapted. Therefore, other instruments are used to equip the empirical investigation based evaluation. A case study is given prior to the investigation to provide a scenario as a basis for the requirements to be elicited and written using the tool-based boilerplate. In this research, a library system is referred to as a study case. Besides, the tool itself is installed in a laptop to be used by the expert. This tool is a prototype developed for the usage of producing an SRS for an information management system. Basic functions of an information management system are mainly the same and can simply adapt to another system. Finally, a simple questionnaire is used to aid the experts to record their experts’ judgment. The questionnaire covers the three quality attributes namely completeness, correctness and consistency to be answered by the experts following the usage of the tool. The three quality attributes were selected because they represent main qualities that lead to a good SRS. Figure 2 depicts the questions used to gather the evaluation results.

C. The Protocol

The environment was set up like an interview session as meeting with experts were made individually at their office. A briefing on the research purpose and explanation on the tool-based boilerplate was made to the expert. Then, about 15 minutes was given to the expert to read the scenario description. The scenario description represented an input to the requirements engineering process as if information gathered earlier in order to write a requirement specification. It basically describes stakeholders’ requirements for the system to be developed. Then, the tool-based boilerplate is given to the expert to write requirements specification based on the scenario description given. Once done, the experts were required to provide feedback based on the questions in Figure 2. For each question, the experts need to score each attribute based on the 5 points Likert scale whereby 1 strongly disagrees and 5 strongly agrees. The entire process will take no longer than 2 hours as the software project scope is small.

D. The Results and Analysis

All the ten experts’ judgment were gathered and analyzed. The judgment was about if the boilerplate technique assists in achieving (or not) the specific quality attributes namely comprehensibility, correctness and consistency within the SRS by implementing the boilerplate technique. The definition of the quality attributes [15] are as listed below:

- **Comprehensibility** – Comprehensibility is achieved if the reader can easily comprehend the meaning of the requirements with minimum explanation.
- **Correctness** - Correctness is achieved if every requirement stated in SRS is one that the software shall meet.
- **Consistency** - Consistency is achieved if and only if no subset of individual requirements stated conflict.

The implementation of the technique was made easy with the assistance of the tool which is referred to as a tool-based boilerplate technique in this paper. Table 2 summarizes the evaluation results based on the ten experts’ judgment. The mean score of comprehensibility is 3.9 while correctness and consistency are both have mean of 4.0.

Table 2
Descriptive Statistics for Comprehensibility, Correctness and Consistency.

	Comprehensibility	Correctness	Consistency
Mean	3.900	4.000	4.000
Median	4.000	4.000	4.000
Mode	4.000	4.000	4.000
Std. Deviation	0.3162	0.6667	0.4714

Figure 3 shows the frequency of the data. As for comprehensibility, nine experts score 4 (Agree) and one expert score 3(Neutral). Besides, two experts score 5 (Strongly Agree), six experts score 4 (Agree) and two experts score 3 (Neutral) for correctness. While for consistency, one expert score 5(Strongly Agree) and 3(Neutral) respectively and the remaining eight experts score 4 (Agree). The mean and the data frequency show that most of the experts at least agree that the boilerplate technique improves the comprehensibility, correctness and consistency of the requirements statements in the SRS.

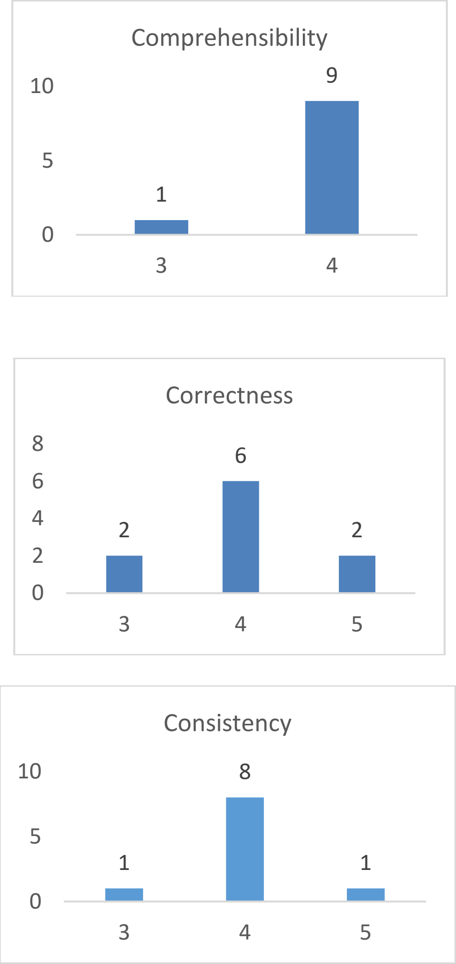


Figure 3: Frequency of data for Comprehensibility, Correctness and Consistency.

V. Conclusion

This paper presents an evaluation of a boilerplate technique to improve SRS quality in comprehensibility, correctness and consistency. The evaluation is done through experts' judgment method in the empirical environment set up. Prior to the evaluation itself, the experts practically walkthrough a simplified requirements engineering process with the assistance of the tool-based boilerplate technique to write functional requirements in the SRS. The experts then evaluated the SRS quality. Theoretically, the boilerplate technique improves the SRS quality in comprehensibility due to guided sentence structure based on generic essential system functionality (information management system) which helps to reduce ambiguity. Besides, the correctness is achieved due to auto-generated wireframe interfaces to indicate exact input, output and potential layout of specific functions. The consistency is achieved with the assistance of the boilerplate uniform sentence structure and its distinct prioritization with the terms 'shall', 'should' and 'will'. The evaluation done by the ten experts agreed that the boilerplate technique improves all the three SRS quality attributes.

Acknowledgment

The research and the publication of this paper are funded by Universiti Teknikal Malaysia Melaka through a research grant PJP/2017/FTMK-CACT/S01573.

References

[1] IEEE Recommended Practice for Software Requirements Specifications. IEEE Standard 830-1998. IEEE Computer Society.

[2] S. Ahmad, I.E.A. Jalil, S.S.S. Ahmad, "An Enhancement of Software Requirements Negotiation with Rule-based Reasoning: A Conceptual Model," Journal of Telecommunication, Electronic and Computer Engineering, vol. 8, no.10, pp. 193-198, 2016.

[3] M. Ortel, M. Malot, A. Baumgart, J.S. Becker, R. Bogusch, S. Farfeleder, N. Gerber, O. Haugen, S. Häusler, B. Josko, J. Mansell, "Requirements Engineering" in CESAR-Cost-efficient Methods and Processes for Safety-relevant Embedded Systems, A. Ajitha, T. Wahl, eds., Springer, pp. 69-143, 2013.

[4] S. Farfeleder, T. Moser, A. Krall, H. Zojer, and C. Panis, "DODT : Increasing Requirements Formalism using Domain Ontologies for Improved Embedded Systems Development," 2011. Acessed from https://www.isis.tuwien.ac.at/node/15540.

[5] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak, "EARS (Easy Approach to Requirements Syntax)," Proc. IEEE Int. Conf. Requirements. Engineering, pp. 317–322, 2009.

[6] C. Arora, M. Sabetzadeh, L. C. Briand, and F. Zimmer, "Requirement Boilerplates : Transition from Manually-Enforced to Automatically-Verifiable Natural Language Patterns," pp. 1–8, 2014

[7] N. Ibrahim, W. M. N. Wan Kadir, and S. Deris, "Documenting requirements specifications using natural language requirements boilerplates," Proceedings of 8th. Malaysian Software Engineering Conference, pp. 19–24, 2014.

[8] N.A. Arbain, M.S. Azmi, L.B. Melhem, A.K. Muda, H. Rashaideh, "Enhancement of Triangle Coordinate for Triangle Features for Better Classification," Jordanian Journal of Computers and Information Technology vol. 2, no. 2, pp. 107-118, 2016.

[9] O. Daramola, G. Sindre, and T. Stalhane, "Pattern-Based Security Requirements Specification Using Ontologies and Boilerplates," pp. 54–59, 2012.

[10] U. Anuar, S. Ahmad, N.A. Emran, "A simplified systematic literature review: Improving Software Requirements Specification quality with boilerplates," Proc. 9th Malaysian Software Engineering Conference (MySEC), 2015, IEEE, pp. 99-105, 2015.

[11] [11] N.A. Emran, "Data completeness measures," Pattern Analysis, Intelligent Security and the Internet of Things, Springer, Cham, pp. 117-130, 2015.

[12] D. Firesmith: "Specifying Good Requirements," Journal of Object Technology, vol. 2, no. 4, pp. 77-87, 2003.

[13] S. Ahmad, "Measuring the Effectiveness of Negotiation in Software Requirements Engineering," University of Western Australia, 2012.

[14] D. Zowghi, & V. Gervasi, "The Three Cs of Requirements: Consistency, Completeness, and Correctness," Proceedings of 8th International Workshop on Requirements Engineering: Foundation for Software Quality, pp. 155–164, 2002.

[15] S. Ahmad and S.A. Asmai, "Measuring Software Requirements Quality Following Negotiation through Empirical Study," International Journal of Applied Engineering Research, vol. 11, no. 6, pp. 4190-4196, 2016.