早稲田大学大学院 基幹理工学研究科

博 士 論 文 概 要

論 文 題 目

Generalized Software Reliability Model Considering Uncertainty and Dynamics: Theoretical Foundations and Empirical Applications

不確実性と時間変化を考慮した一般化ソフトウェア信頼性モデル：理論的基礎と実証的応用

申 請 者

Kiyoshi	HONDA
本田	澄

情報理工学専攻 高信頼ソフトウェア工学研究

2016 年 12 月

Software reliability is a critical component of computer system availability. Software reliability growth models (SRGMs), such as the Times Between Failures Model and Failure Count Model, can indicate whether a sufficient number of faults have been removed to release the software. The Failure Count Model is based on counting failures and probability methods. Representatives of this type of model include the Goel-Okumoto NHPP Model and the Musa Execution Time Model.

Recent studies have attempted to describe the dynamics of developments using a stochastic process, however they have not evaluated the dynamics with actual dataset. Existing methods assume that each parameter is independent of time, which leads to the inability to account for dynamics. I hypothesize that if several developers suddenly join a project in which a SRGM is applied, the development environment suddenly changes, and resulting in several influences, especially on the SRGM parameters since such sudden changes should be treated with time dependent parameters. These assumptions limit the models but also make the models solvable by mathematical methods. For example, the Goel-Okumoto NHPP model has two parameters (i.e., the total number of faults and fault detection rate), which are independent of time because the Goel-Okumoto NHPP model equations cannot be solved if these values have time dependencies. Although Okamura et al. propos a multi-factor software reliability model framework which can deal with the measurements observed in testing phase such as test coverage and the number of test workers and so on. Their method employ the logistic regression to analyze the relationship between the probability of occurrence of an event and environmental factors which would include person hours. Although they evaluate their method with five datasets including the number of test cases, cumulative test cases, and increment of code coverage and so on, they do not evaluate their method with person hours.

On the other hand, software development projects have uncertainties such as developers’ skills and development environments. However, existing software reliability growth models do not contain these uncertainty elements. I assume that the uncertainties in the development process affect the ability to detect faults and are modeled as a Gaussian white noise that is a simple but commonly used noise.

As far as I know, previous studies use linear stochastic differential equations, but my research uses nonlinear stochastic differential equations to model actual situations more realistically. These studies and SRGMs treat several datasets only in the given situation (e.g., within the same company or organization). In short, existing models are applied to the situation from which the datasets are obtained, and are evaluated from different domains. For example, one work evaluates several SRGMs with automotive software datasets, while another assess two SRGMs with an army system dataset. Rana et al. study four software projects from the automotive sector and conclude two statistic SRGMs perform better than other SRGMs. On the other hand, Goel et al. investigate two stochastic SRGMs with a U.S. Navy project dataset and conclude their model provides a plausible description. These studies do not evaluate existing SRGMs with other domains.

SRGMs sometimes misfit the actual data in ongoing developments. In addition, the results do not always match the developers' expectations. If SRGMs misfit the actual data, the managers and developers will decide wrong plans, for example stopping testing early or release software which has not been tested enough. On the other hand, if the results of SRGMs indicate that the faults are enough detected or not enough detected contrary to the managers and developers' expectations, they will decide wrong plans too. If software is released with several faults left, the company which have released it will take time to debug it.

In order to treat the dynamics and uncertainties in software developments, I propose a model called the Generalized Software Reliability Model (GSRM) to describe several development situations that involve random factors (e.g., developers’ skills and development environment) to estimate the time that a development will end. Additionally, I predict the release times of open source software (OSS) using the GSRM and agile development. Moreover, I apply the GSRM and SRGMs to company's datasets and predict the ranges of the release times. Due to random factors, the GSRM in each situation has an upper and lower limit, suggesting that the GSRM can predict the maximum and minimum number of faults. I formulate the upper and lower limit equations for three development situations with approximations in order to treat these equations easily and predict the ranges of the release times.

I evaluate the GSRM and other models using datasets from different organizations and circumstances. The GSRM can quantify uncertainties that are influenced by random factors, which is important to more accurately model the growth of software reliability and to optimize development teams or environments. Additionally I apply the GSRM to companies’ dataset and propose applications of the GSRM.

In my thesis, I explain about the GSRM which treats uncertainties and dynamics and propose its applications. In Chapter 4, I define the methods to predict release times. In Chapter 5, I define the ranges of release time. In Chapter 6, I apply SRGMs to the datasets of two projects in order to determine when SRGMs provide ill-fitted or unexpected results. In Chapter 7, in order to avoid misfitting and mismatching the developers' expectations, I propose good accuracy SRGMs using person hours. Additionally I propose a leveled SRGM which indicates a standard line of developments and can be used to help managers and developers assess the progress of a development. In Chapter 8, I extend the leveled SRGMs by classifying the projects.

My thesis is organized as follows.

In Chapter 1, I introduce the objective of this research with research background. This chapter defines the research area of the thesis by referring related works.

In Chapter 2, I describe the overview of this thesis.

In Chapter 3, to evaluate my model, I simulate nine types of development situations using the Monte Carlo method. To simplify the application of my model, I divide the situations related to uncertainty situations into three types and derive an approximation equation for each. Finally, I apply the approximation equations to four projects from two different organizations and classify these projects into three uncertainty types.

In Chapter 4, I define the methods to predict release times by using the GSRM and an NHPP model (Exponential model). In general, the time when the development will end is planned as when the number of detected faults is getting around 95% of the predicted number of faults. SRGMs can calculate the predicted number of faults and the time when the development will end. Using the GSRM, I successfully predict the release dates and the number of issues about OSS.

In Chapter 5, I define uncertainty values from actual data containing information on the faults during development, and apply the GSRM to three datasets to calculate the ranges of time, including the range of possible development times considering the uncertainty values. I also examined how well the GSRM can predict the required development time using actual datasets. By using past data, the GSRM can calculate the uncertainties and predict how long a project will take.

In Chapter 6, I apply SRGMs to the datasets of two projects developed by Fujitsu Labs Ltd. in order to determine when SRGMs provide ill-fitted or unexpected results. I assume that the detected faults differ by test phase, and this difference is the source of misfit between the model and actual data. To investigate the source of unexpected results, two different SRGMs are used. Separating the faults into test phases and combining the results provides a better fitting model. I successfully obtain a good fitness model by separating faults by test phase and applying SRGMs. Moreover, I find unexpected situations in development by monitoring the faults and the behavior of SRGMs. These results demonstrate that if developers and managers monitor the behavior of the SRGM results from the beginning of development, they can detect several unexpected situations earlier than ever.

In Chapter 7, in order to avoid misfitting and mismatching the developers' expectations, I propose new good accuracy SRGMs using person hours. I assume that SRGMs based on calendar time cannot realize accurate predictions, because many kinds of SRGMs treat calendar time, which includes holidays and non-testing time and does not reflect the actual efforts by developers. Finally I successfully model nine actual datasets. SRGMs based on person hours are between 13% and 97% more precise than those based on calendar time. Moreover, I propose leveled SRGMs based on the fault density and the rates of person hours as well as the rates of used test cases. The leveled SRGMs indicate the standard line of development, which can be used to help managers and developers assess the progress of a development. My interviews of managers about the models indicate that leveled SRGMs provide useful information about the progress of the projects.

In Chapter 8, I extend the leveled SRGM by classifying the projects contained within the leveled SRGM. Prior to the test phases, I select system scale parameters such as the lines of code, number of test cases, and test density (which is defined as the number of test cases divided by lines of code) as classification parameters to create a leveled SRGM. I successfully model nine actual datasets by classifying with system scale parameters. The SRGM classified by test density can more precisely model the data than other classifications, including no classification.

In Chapter 9, I conclude the thesis and explain future work.

早稲田大学　博士（工学）　学位申請　研究業績書

氏 名　本田 澄　　印

(2016 年　12 月　現在)

種 類 別	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）
1.論文 [1] ○	Generalized Software Reliability Model Considering Uncertainty and Dynamics: Model and Applications, Kiyoshi Honda, Hironori Washizaki and Yoshiaki Fukazawa International Journal of Software Engineering and Knowledge Engineering (IJSEKE), pp.1-28, Sep. 2017. (掲載決定)
[2]	まねっこダンス:真似て覚えるプログラミング学習ツール, 坂本 一憲, 本田 澄, 音森 一輝, 山崎 頌平, 服部 真智子, 松浦 由真, 高野 孝一, 鷲崎 弘宜, 深澤 良彰, 日本ソフトウェア科学会, 32(4)pp.103-114, Nov. 2015.
2.講演 (国際会議) [1]	Identifying Potential Problems and Risks in GQM+Strategies Models Using Metamodel and Design Principles, Chimaki Shimura, Hironori Washizaki, Takanobu Kobori, Yohei Aoki, Kiyoshi Honda, Yoshiaki Fukazawa, Katsutoshi Shintani and Takuto Nonomura, 50th Annual Hawaii International Conference on System Sciences (HICSS-50), pp.1-10, 2017.
[2]	Evaluating Software Product Quality based on SQuaRE Series, Hidenori Nakai, Naohiko Tsuda, Kiyoshi Honda, Hironori Washizaki, Yoshiaki Fukazawa, IEEE TENCON 2016, pp.1-4, 2016.
[3] ○	Case Study: Project Management Using Cross Project Software Reliability Growth Model Considering System Scale, Kiyoshi Honda, Nobuhiro Nakamura, Hironori Washizaki and Yoshiaki Fukazawa, 27th International Symposium on Software Reliability Engineering (ISSRE 2016), pp.1-4, 2016.
[4]	Initial Framework for a Software Quality Evaluation based on ISO/IEC 25022 and ISO/IEC 25023, Hidenori Nakai, Naohiko Tsuda, Kiyoshi Honda, Hironori Washizaki and Yoshiaki Fukazawa, The 2016 IEEE International Conference on Software Quality, Reliability & Security (QRS 2016), pp. 410-411, 2016.
[5] ○	Case Study: Project Management Using Cross Project Software Reliability Growth Model, Kiyoshi Honda, Nobuhiro Nakamuray, Hironori Washizaki and Yoshiaki Fukazawa, The 2016 IEEE International Conference on Software Quality, Reliability and Security Companion, pp.39-46, 2016.
[6]	GO-MUC: A Strategy Design Method Considering Requirements of User and Business by Goal-Oriented Measurement, Chihiro Uchida, Kiyoshi Honda, Hironori Washizaki, Yoshiaki Fukazawa, Kentaro Ogawa, Tomoaki Yagi, Mikako Ishigaki, Masashi Nakagawa, 9th International Workshop on Cooperative and Human Aspects of Software Engineering, pp. 93-96, 2016.
[7] ○	Software Reliability Growth Model Considering Uncertainty and Dynamics in Development, Kiyoshi Honda, Hironori Washizaki and Yoshiaki Fukazawa, 23rd IEEE

早稲田大学　博士（工学）　学位申請　研究業績書

種 類 別	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）
	International Conference on Software Analysis, Evolution, and Reengineering, pp.1-1, 2016.
[8]	Toward selecting a reliable version of OSS library based on bug-fixing curve, Keisuke Fujino, Akinori Ihara, Kiyoshi Honda, Hironori Washizaki, Kenichi Matsumoto, 23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering (poster), 2016.
[9] ○	Case Study: Software Reliability Growth Model Based on Person Hours, Kiyoshi Honda, Nobuhiro Nakamura, Hironori Washizaki and Yoshiaki Fukazawa, 7th IEEE International Workshop on Empirical Software Engineering in Practice (poster), 2016.
[10] ○	Detection of Unexpected Situations by Applying Software Reliability Growth Models to Test Phases, Kiyoshi Honda, Hironori Washizaki, Yoshiaki Fukazawa, Kazuki Munakatay, Sumie Moritay, Tadahiro Ueharay, and Rieko Yamamoto, Proceedings of the 26th IEEE International Symposium on Software Reliability Engineering, pp.2-5, 2015.
[11] ○	Predicting Release Time for Open Source Software based on the Generalized Software Reliability Model, Hironori Washizaki, Kiyoshi Honda and Yoshiaki Fukazawa, Proceedings of Agile Conference 2015, pp. 76-81, 2015.
[12]	Comparative Study on Programmable Robots as Programming Educational Tools, Shohei Yamazaki, Kazunori Sakamoto, Kiyoshi Honda, Hironori Washizaki and Yoshiaki Fukazawa, Proceedings of the 17th Australasian Computing Education Conference, pp. 155-164, 2015.
[13] ○	Predicting Time Range of Development Based on Generalized Software Reliability Model, Kiyoshi Honda, Hidenori Nakai, Hironori Washizaki, Yoshiaki Fukazawa, Ken Asoh, Kazuyoshi Takahashi, Kentarou Ogawa, Maki Mori, Takashi Hino, Yosuke Hayakawa, Yasuyuki Tanaka, Shinichi Yamada and Daisuke Miyazaki, 21st Asia-Pacific Software Engineering Conference, pp.351-358, 2014.
[14]	Initial Industrial Experience of GQM-based Product-Focused Project Monitoring with Trend Patterns, Hidenori Nakai, Kiyoshi Honda, Hironori Washizaki, Yoshiaki Fukazawa, Ken Asoh, Kaz Takahashi, Kentarou Ogawa, Maki Mori, Takashi Hino, Yosuke Hayakawa, Yasuyuki Tanaka, Shinichi Yamada and Daisuke Miyazaki, 21st Asia-Pacific Software Engineering Conference, pp.43-46, 2014.
[15]	Continuous Product-Focused Project Monitoring with Trend Patterns and GQM, Hidenori Nakai, Kiyoshi Honda, Hironori Washizaki, Yoshiaki Fukazawa, Ken Asoh, Kaz Takahashi, Kentarou Ogawa, Maki Mori, Takashi Hino, Yosuke Hayakawa, Yasuyuki Tanaka, Shinichi Yamada and Daisuke Miyazaki, Proceedings of the 2nd International Workshop on Quantitative Approaches to Software Quality, pp. 69-74,

早稲田大学　博士（工学）　学位申請　研究業績書

種 類 別	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）
	2014.
[16]	Toward Monitoring Bugs-fixing Process after the Releases in Open Source Software, Keisuke Fujino, Akinori Ihara, Kiyoshi Honda, Hironori Washizaki and Kenichi Matsumoto, 6th International Workshop on Empirical Software Engineering in Practice (poster), 2014.
[17] ○	Predicting the Release Time Based on a Generalized Software Reliability Model (GSRM), Kiyoshi Honda, Hironori Washizaki and Yoshiaki Fukazawa, Proceedings of the 38th Annual IEEE International Computers, Software, and Applications Conference, pp.604-605, 2014.
[18] ○	A Generalized Software Reliability Model Considering Uncertainty and Dynamics in Development, Kiyoshi Honda, Hironori Washizaki and YoshiakiFukazawa, Proceedings of 14th International Conference of Product Focused Software Development and Process Improvement, pp.342-346, 2013
2.講演 (国内会議) [1] ○	欠陥とソースコードの変更回数の関係分析，本田 澄，坂口 英司，伊原 彰紀，鷲崎 弘宜，深澤 良彰，情報処理学会ソフトウェア工学研究会ウィンターワークショップ 2016・イン・逗子，pp.57-58，2016.
[2] ○	オープンソースソフトウェアに関するソースコードの変更回数とバグ修正の関係分析に向けて，本田 澄，伊原 彰紀，鷲崎 弘宜，深澤 良彰，日本ソフトウェア科学会 第 22 回 ソフトウェア工学の基礎ワークショップ (poster)，2015.
[3] ○	開発者行動を考慮したソフトウェア信頼性モデル，本田 澄，鷲崎 弘宜，深澤 良彰，情報処理学会ソフトウェア工学研究会ウィンターワークショップ 2015・イン・宜野湾，pp.37-38，2015.
[4] ○	OSS の不具合修正曲線に基づく残存未修正不具合数の予測の試み，藤野 啓輔，伊原 彰紀，本田 澄，鷲崎 弘宜，松本 健一，日本ソフトウェア科学会 第 21 回 ソフトウェア工学の基礎ワークショップ，pp.57-62，2014.
[5] ○	開発者数の変動を含むソフトウェア信頼性モデル，本田 澄，中井 秀矩，鷲崎 弘宜，深澤 良彰，日本ソフトウェア科学会 第 21 回 ソフトウェア工学の基礎ワークショップ 2014 (poster)，2014.
[6] ○	不確実性を含む信頼性成長モデル，本田 澄，鷲崎 弘宜，深澤 良彰，情報処理学会ウィンターワークショップ 2014・イン・大洗，pp.15-16，2014.
[7]	GQM を用いた改善プロセスサポートツールの開発，中井 秀矩，本田 澄，鷲崎 弘宜，深澤 良彰，情報処理学会ウィンターワークショップ 2014・イン・大洗，pp.105-106，2014.
	その他 10 件