Using Source Code Metrics and Multivariate Adaptive Regression Splines to Predict Maintainability of Service Oriented Software

Lov Kumar[1] Santanu Kumar Rath[1] Ashish Sureka[2]

[1]NIT Rourkela, India
Email: lovkumar505@gmail.com, skrath@nitrkl.ac.in

[2]ABB India, India
Email: ashish.sureka@in.abb.com

HASE 2017

Table of Contents

Maintainability Prediction - Service Oriented Software

Service-Oriented Computing (SOC) paradigm is the dominant development platform for software systems

One most effective way to control the maintenance costs is to utilize software metrics during the development phase

Web Services are observed to be described using a WSDL (Web Service Description Language) document

Maintainability prediction for web service interfaces and WSDL documents is relatively unexplored

Research Aim

Aim 1: To investigate the usefulness of metrics for the source code implementing a web service for maintainability prediction

Aim 2: To examine the application of multivariate adaptive regression splines (MARS) method to build estimators based on source code metrics as predictor variables and maintainability as target variable

Aim 3: To compare the performance of MARS model with performance of the model developed using multivariate linear regression models (MLR), and support vector machine (SVM) methods

Table of Contents

Literature Survey and Analysis

Bingu Shim et al. [131]

Authors observed that quality attributes depends on various number of design properties, i.e., consumability, cohesion, coupling, complexity, size, different types of granularity. They derive metrics and compute the relation between design metrics and quality

Cristian Mateos et al. [100]

Authors work on the hypothesis that object-oriented source code metrics avoid the occurrence of anti-pattern. They considered twelve different types of source code metrics such as: AMC, CAM, CBO, WMC, LOC, LCOM, LCOM3, RFC, EPM, VTC, ATC, DAM to find the anti-pattern occurrence at the WSDL level

Literature Survey and Analysis

Mikhail Perepletchikov et al. [119][120]

Authors propose cohesion and coupling metrics for predicting maintainability parameters of Service-Oriented Software [119] [120]. Authors redefined the cohesion and coupling metrics for service-oriented software

Yu, Yijun [148]

OO source code metrics are being used for quality indicators of SOA system [148] [100] [39]

Table of Contents

Unique and Novel Contributions

[1] Development of Data Collection and Preparation Process

[2] Development of a maintainability prediction model for service oriented application using MARS and source code metrics

[3] Selection of right set of source code metrics to improve the performance of maintainability prediction model

[4] Empirical analysis to demonstrate the effectiveness of the proposed approach

Table of Contents

Source Code Metrics

11 Different Source Code Metrics

1. Depth of inheritance tree (DIT)
2. Weighted method per class (WMC)
3. Coupling between object (CBO)
4. Number of children (NOC)
5. Number of methods (NOM)
6. Lack of cohesion among methods (LCOM)
7. Response for class (RFC)

Source Code Metrics

11 Different Source Code Metrics

8 Data abstraction coupling (DAC)

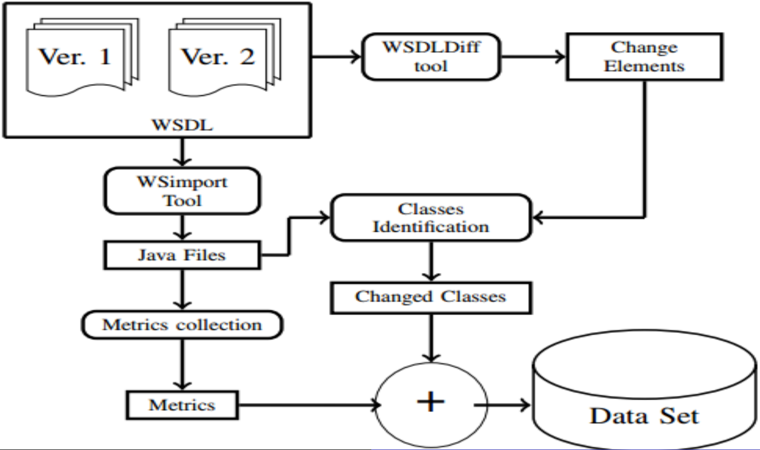
9 SIZE2

10 SIZE1

11 McCabe Cyclomatic complexity (MVG)

Source code metrics are computed using tool such as CKJM and LOC metrics. These metrics include the popular metrics suite proposed by Chidamber and Kemerer [35], and Li and Henry [92]

Data Collection and Preparation Process



Data Collection and Preparation Process

Step 1: WSDLDiff Tool

WSDLDiff tool is used for comparing subsequent versions of WSDL interfaces ([38]) and finding the name (Operation, Message etc.) and type (addition, move, removed, update etc.) of the elements affected by the changes

Step 2: WSimport Tool

Wsimport tool is used to parse WSDL document of web-service and generate Java files

Data Collection and Preparation Process

Step 3: Metrics Collection

Metrics values of source code are computed using tools such as CKJM and LOC metrics

Step 4: Class Identification

The Java classes which contain the changed elements (Operation, Message etc.) are termed as changed classes. These types of classes containing changed elements are identified

Step 5: Data Set

The change statistics from Step 4 and metrics from Step 3 are combined to generate dataset for further processing

eBay Web Service

Version	# classes	LOC	# changed classes	# non-changed classes	% changed classes
863	1507	1320264	175	1332	11.61
865	1507	1320576	102	1405	6.77
867	1524	1335460	240	1284	15.75
869	1509	1322904	128	1381	8.55
871	1509	1323080	119	1390	7.89

The number of classes for all the five versions are more than 1500. The percentage of changed classes from eBay version 863 to eBay version 865 is 11.61 and the the percentage of change classes from 865 to 867 is 6.77

Table of Contents

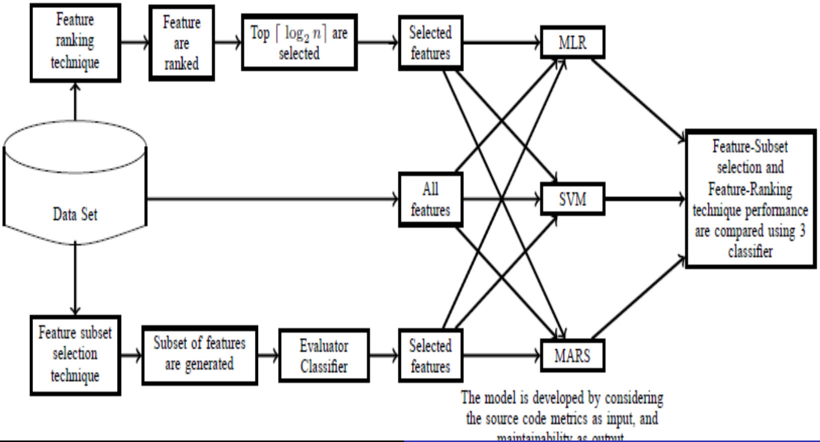
Model Variables

Analysis	Dependent Variable	Independent Variable
A1	Maintainability	DIT, WMC, NOC, RFC, CBO, NOM, LCOM, MVG, DAC, SIZE1, SIZE2
A2	Maintainability	Reduced feature attributes using feature ranking techniques
A3	Maintainability	Extracted feature attributes using feature subset selection techniques

Maintainability or change is considered as a dependent variable and set of source code metrics as a independent variables respectively

Table of Contents

Research Methodology



Research Methodology

Four types of FR techniques and four types of FSS techniques to select optimal set of source code metrics. These all selected set of source code metrics are applied on five different versions of eBay dataset

A total of 135 ((1 considering all features+ 8 feature selection technique) * 5 datasets * 3 different classification technique) distinct classification models are considered in the study

Research Methodology

Step 1

Eleven different source code metrics from the bytecode of the compiled Java files of web service are computed

Step 2

Four different FR techniques are applied on all five versions of eBay web service. Each FR technique uses some parameter to sort the features and further top $\lceil \log_2 n \rceil$ ranked features out of n features are used as input to develop a model

Research Methodology

Step 3

Four different FSS techniques have been considered to select optimal set of features

Step 4

All elven source code metrics, selected set of source code metrics using feature selection techniques are validated using three different classification methods i.e., MLR, SVM, and MARS

Naming Conventions for different Techniques

Abbreviation	Corresponding Name
AM	All Metrics
FR1	CS test
FR2	GR Feature Evaluation
FR3	IG Feature Evaluation
FR4	OneR Feature Evaluation
FS1	Classifier Subset Evaluation
FS2	Correlation based Feature Selection
FS3	Filtered Subset Evaluation
FS4	Rough Set Analysis (RSA)

Table of Contents

Selected metrics after Feature ranking methods

Method	863	865	867	869	871
FR1	SIZE1, MVG, DAC, SIZE2	DAC, MVG, CBO, LCOM	SIZE1, CBO, DAC, MVG	DAC, LOCM, MVG, SIZE1	DAC, LOCM, WMC, CBO
FR2	DAC, WMC, LCOM, RFC	NOM, WMC, LOCM, DAC	NOM, DIT, MVG, SIZE2	NOM, DIT, RFC, WMC	LOCM, NOM, WMC, RFC
FR3	SIZE1, MVG, SIZE2, CBO	DAC, MVG, LCOM, WMC	SIZE1, RFC, MVG, CBO	DAC, LCOM, MVG, SIZE1	LOCM, WMC, DAC, MVG
FR4	DAC, MVG, NOC, NOM	DAC, MVG, NOC, NOM	MVG, DAC, NOM, LCOM	DAC, MVG, NOC, NOM	DAC, NOC, DIT, CBO

Observations and Results

In case of MLR, model developed by considering selected set of source code metrics using classifier subset evaluation as input produces better result as compared to others.

In case of SVM, model developed by considering selected set of source code metrics using correlation based feature selection as input produces better result as compared to others.

In case of MARS, model developed by considering selected set of source code metrics using classifier subset evaluation as input produces better result as compared to others.

Table of Contents

Selected metrics after Feature subset selection

Method	863	865	867	869	871
FS1	WMC, DIT, DAC, MVG	WMC, DIT, DAC	DIT, CBO, LCOM, DAC, NOM,MVG	DIT,DAC, NOM	DIT, LCOM,DAC
FS2	DIT, DAC, WMC, SIZE1, NOC	DIT, DAC, WMC, SIZE1, NOC	DIT, DAC, WMC, SIZE1, NOC	DIT, DAC, WMC, SIZE1, NOC	DIT, DAC, WMC, SIZE1, NOC
FS3	DIT, MVG, WMC, DAC	DAC, DIT, WMC	DIT, MVG, LCOM, NOM, CBO, DAC	NOM, DAC, DIT	DAC, DIT, LCOM
FS4	DIT, MVG, DAC, NOM, RFC, SIZE2, SIZE1	DIT, MVG, DAC, NOM, RFC, SIZE2, SIZE1	DIT, MVG, DAC, NOM, RFC, SIZE2, SIZE1	DIT, MVG, DAC, NOM, RFC, SIZE2, SIZE1	DIT, MVG, DAC, NOM, RFC, SIZE2, SIZE1

Observations and Results

In case of MLR, model developed by considering selected set of source code metrics using classifier subset evaluation as input produces better result as compared to others

In case of SVM, model developed by considering selected set of source code metrics using correlation based feature selection as input produces better result as compared to others

In case of MARS, model developed by considering selected set of source code metrics using classifier subset evaluation as input produces better result as compared to others

Multivariate Adaptive Regression Splines (MARS)

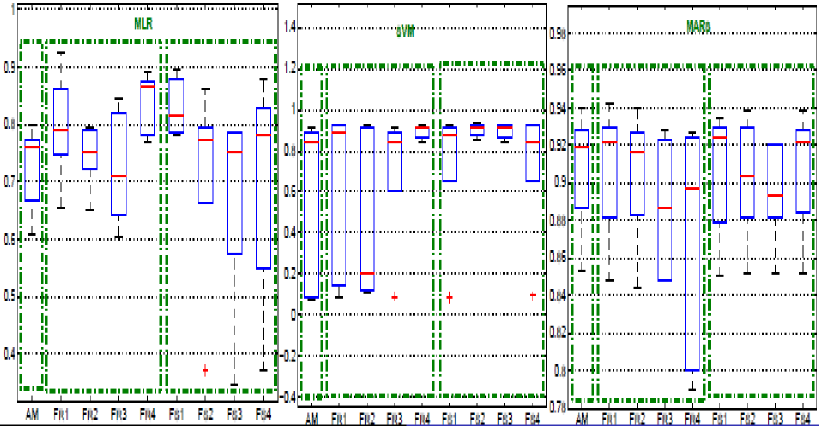
The hardware used to carry out our experiments are: Core i5 processor with 4GB RAM and storage capacity of 250GB hard disk

Prediction models are developed using the licensed MATLAB environment at NIT-Rourkela

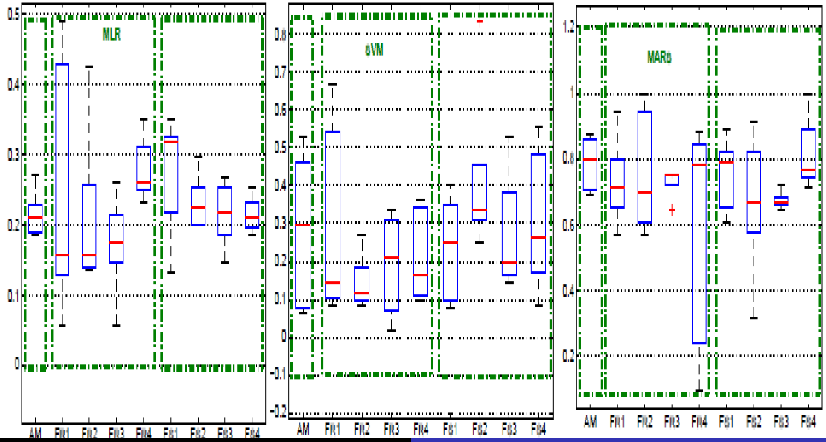
Model developed for predicting web-service maintainability using MARS yields better result as compared to MLR, and SVM methods

Table of Contents

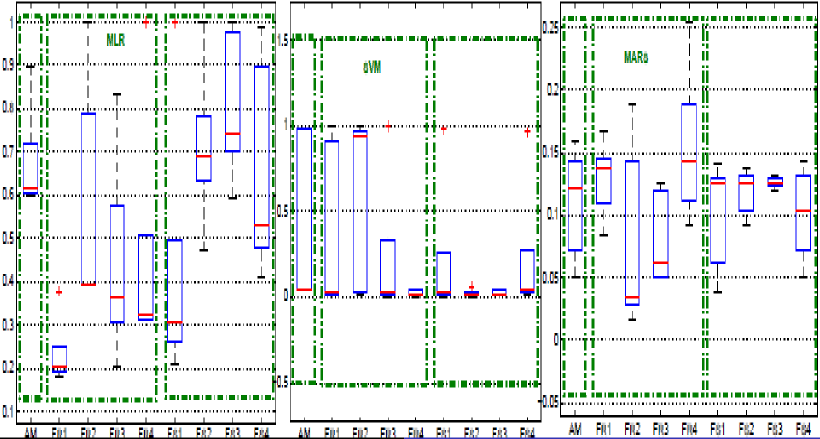
Accuracy



Precision



Recall



Overall Performance

In case of MLR method, model developed by considering selected set of source code metrics using oneR feature evaluation as input produces better result as compared to others

In case of SVM, model developed by considering selected set of source code metrics using correlation based feature selection as input produces better result as compared to others

In case of MARS, model developed by considering selected set of source code metrics using classifier subset evaluation as input produces better result as compared to others

t-test: Among Different Feature Selection Techniques

Mean Difference									
	AM	FR1	FR2	FR3	FR4	FS1	FS2	FS3	FS4
AM	0	-0.04	0.03	-0.04	-0.14	-0.09	-0.11	-0.09	-0.04
FR1	0.04	0	0.07	0	-0.1	-0.05	-0.07	-0.05	-0.01
FR2	-0.03	-0.07	0	-0.07	-0.16	-0.12	-0.13	-0.12	-0.07
FR3	0.04	0	0.07	0	-0.1	-0.05	-0.07	-0.05	0
FR4	0.14	0.1	0.16	0.1	0	0.04	0.03	0.05	0.09
FS1	0.09	0.05	0.12	0.05	-0.04	0	-0.02	0	0.05
FS2	0.11	0.07	0.13	0.07	-0.03	0.02	0	0.02	0.06
FS3	0.09	0.05	0.12	0.05	-0.05	0	-0.02	0	0.05
FS4	0.04	0.01	0.07	0	-0.09	-0.05	-0.06	-0.05	0
p-value									
	AM	FR1	FR2	FR3	FR4	FS1	FS2	FS3	FS4
AM	-	0.64	0.76	0.5	0.1	0.37	0.2	0.3	0.49
FR1	0.64	-	0.22	0.95	0.19	0.57	0.4	0.53	0.92
FR2	0.76	0.22	-	0.33	0.06	0.26	0.14	0.21	0.34
FR3	0.5	0.95	0.33	-	0.11	0.52	0.28	0.46	0.92
FR4	0.1	0.19	0.06	0.11	-	0.46	0.35	0.16	0.15
FS1	0.37	0.57	0.26	0.52	0.46	-	0.82	0.96	0.58
FS2	0.2	0.4	0.14	0.28	0.35	0.82	-	0.62	0.27
FS3	0.3	0.53	0.21	0.46	0.16	0.96	0.62	-	0.48
FS4	0.49	0.92	0.34	0.92	0.15	0.58	0.27	0.48	-

Additional t-test Results

t-test: Among different Classifier

	Mean Difference			p-value			t-value		
	MLR	SVM	MARS	MLR	SVM	MARS	MLR	SVM	MARS
MLR	0	0.03	-0.15	-	0.58	0	-	0.56	-8.13
SVM	-0.03	0	-0.18	0.58	-	0	-0.56	N-	-3.61
MARS	0.15	0.18	0	0	0	-	8.13	3.61	NaN

t-test: feature ranking Versus feature subset selection techniques (Average)

Mean Difference	p-value	t-value
-0.0362	0.36	-0.9226

t-test Analysis

We use pairwise t-test to compare the performance of feature selection techniques and classification techniques

One part of the result table shows the p-value and the other part shows the mean difference values of performance parameter

Pairwise t-test has been employed to determine whether feature ranking methods work better than feature subset selection methods or both have performed equally well

Final Conclusion and Takeaways

Experimental analysis reveals existence of a small set of source code metrics from a large number of available source code software metrics across various types which are able to accurately predict maintainability with few mis-classification errors

The web-service maintainability prediction model developed using MARS shows better results in comparison to MLR and SVM techniques

We can infer that oneR feature evaluation yields better results compared to other techniques

Final Conclusion and Takeaways

There is a significant difference between different classification techniques. According to the value of mean difference, MARS yields better result compared to other techniques.

The performance of the feature selection methods is varied with the different classification methods used

References I

Golnoush Abaei, Ali Selamat, and Hamido Fujita.
An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction.
Knowledge-Based Systems, 74:28–39, 2015.

F. B. E Aberu.
The mood metrics set.
In *Proceedings of ECOOP'95 workshop on metrics*, Europe, 1995.

F. B. E. Abreu and R. Carapuca.
Object-Oriented software engineering: Measuring and controlling the development process.
In *Proceedings of the 4th International Conference on Software Quality*, volume 186, pages 1–8, 1994.

Hojjat Adeli and Shih-Lin Hung.
Machine learning: neural networks, genetic algorithms, and fuzzy systems.
John Wiley & Sons, Inc., 1994.

References II

KK Aggarwal, Yogesh Singh, Pravin Chandra, and Manimala Puri.
Measurement of software maintainability using a fuzzy model.
Journal of Computer Science, 1(4):538–542, 2005.

KK Aggarwal, Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra.
Application of artificial neural network for predicting maintainability using object-oriented metrics.
Transactions on Engineering, Computing and Technology, 15:285–289, 2006.

KK Aggarwal, Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra.
Empirical analysis for investigating the effect of object-oriented metrics on fault proneness: a replicated case study.
Software process: Improvement and practice, 14(1):39–62, 2009.

Briand L.C Arisholm E and Johannessen E.B.
A systematic and comprehensive investigation of methods to build and evaluate fault prediction models.
Empirical Software Engineering, 83(1):2–17, 2010.

References III

J Scott Armstrong and Fred Collopy.
Error measures for generalizing about forecasting methods: Empirical comparisons.
International Journal of Forecasting, 8(1):69–80, 1992.

Rajendra K. Bandi, Vijay K. Vaishnavi, and Daniel E Turk.
Predicting maintenance performance using object-oriented design complexity metrics.
IEEE Transactions on Software Engineering, 29(1):77–87, 2003.

Rajiv D Banker, Srikant M Datar, Chris F Kemerer, and Dani Zweig.
Software complexity and maintenance costs.
Communications of the ACM, 36(11):81–94, 1993.

Rajiv D Banker, Srikant M Datar, Chris F Kemerer, and Dani Zweig.
Estimation and prediction metrics for adaptive maintenance effort of object-oriented systems.
IEEE Transactions on Software Engineering, 27(12):1062–1084, 2001.

References IV

J. Bansiya and C. G. Davis.
A hierarchical model for Object-Oriented design quality assessment.
ACM Transactions on Programming Languages and Systems., 128(1):4–17, August 2002.

V. R. Basili, L. C. Briand, and W. L. Melo.
A validation of Object-Oriented design metrics as quality indicators.
IEEE Transactions on Software Engineering, 22(10):751–761, October 1996.

Victor R Basili, Lionel C Briand, and Walcélio L Melo.
How reuse influences productivity in object-oriented systems.
Communications of the ACM.

Dilek Baski and Sanjay Misra.
Metrics suite for maintainability of extensible markup language web services.
IET Software, 5(3):320–341, 2011.

References V

R Battiti.
First and second-order methods for learning between steepest descent and newton's method.
Neural Computation, 4(2):141–166, 1992.

Aaron B Binkley and Stephen R Schach.
Validation of the coupling dependency metric as a predictor of run-time failures and maintenance measures.
In *Proceedings of the 20th international conference on Software engineering*, pages 452–455. IEEE Computer Society, 1998.

Barry W. Boehm.
Software engineering economics.
IEEE Transactions on Software Engineering, 10(1):4–21, 1984.

Janez Brank, Marko Grobelnik, Natasa Milic-Frayling, and Dunja Mladenic.
Feature selection using linear support vector machines.
Microsoft Research, Microsoft Corporation, pages 1–8, 2002.

References VI

Janez Brank, Marko Grobelnik, Natasa Milic-Frayling, and Dunja Mladenic.
Consistency-based search in feature selection.
Artificial intelligence, 151(1):155–176, 2003.

L. C. Briand and J. W*ü*st.
Modeling development effort in Object-Oriented systems using design properties.
IEEE Transactions on Software Engineering, 27(11):963–986, November 1993.

L. C. Briand, J. W*ü*st, J. W. Daly, and D. V. Porter.
Exploring the relationships between design measures and software quality in Object-Oriented systems.
The Journal of Systems and Software, 51(3):245–273, May 2000.

Lionel C Briand, Jürgen Wüst, John W Daly, and D Victor Porter.
Exploring the relationships between design measures and software quality in object-oriented systems.
Journal of systems and software, 51(3):245–273, 2000.

References VII

Lionel C Briand, Jürgen Wüst, Stefan V Ikonomovski, and Hakim Lounis.
Investigating quality factors in object-oriented designs: an industrial case study.
In *Proceedings of the 21st international conference on Software engineering*, pages 345–354. ACM, 1999.

D. S. Broomhead and David Lowe.
Multivariable functional interpolation and adaptive networks.
Complex Systems, 2:321–355, 1988.

C.J. Burgess and M.Lefley.
Can genetic programming improve software effort estimation.
Information and Software Technology, 43:863–873, 2001.

Rachel Burrows, Fabiano C Ferrari, Otavio AL Lemos, Alessandro Garcia, and Francois Taiani.
The impact of coupling on the fault-proneness of aspect-oriented programs: an empirical study.
In *Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on*, pages 329–338, 2010.

References VIII

Jones C.
Software quality in 2010: a survey of the state of the art.
In *Founder and Chief Scientist Emeritus*, 2010.

Massimo Carbone and Giuseppe Santucci.
Fast&&serious: a uml based metric for effort estimation.
In *Proceedings of the 6th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE02)*, pages 313–322, 2002.

Michelle Cartwright and Martin Shepperd.
An empirical investigation of an object-oriented software system.
IEEE Transactions on Software Engineering, 26(8):786–796, 2000.

Jie-Cherng Chen and Sun-Jen Huang.
An empirical analysis of the impact of software development problem factors on software maintainability.
Journal of Systems and Software, 82(6):981–992, 2009.

References IX

S. R. Chidamber and C. F. Kemerer.
A metrics suite for Object-Oriented design.
IEEE Transactions on Software Engineering, 20(6):476–493, 1994.

Shyam R Chidamber, David P Darcy, and Chris F Kemerer.
Managerial use of metrics for Object-Oriented software: An exploratory analysis.
IEEE Transactions on Software Engineering, 24(8):629–639, 1998.

Shyam R Chidamber and Chris F Kemerer.
Towards a metrics suite for Object-Oriented design, volume 26.
ACM.

Don Coleman, Dan Ash, Bruce Lowther, and Paul Oman.
Using metrics to evaluate software system maintainability.
IEEE Computer, 27(8):44–49, 1994.

Don Coleman, Bruce Lowther, and Paul Oman.
The application of software maintainability models in industrial software systems.
Journal of Systems and Software, 29(1):3–16, 1995.

References X

José Luis Ordiales Coscia, Marco Crasso, Cristian Mateos, Alejandro Zunino, and Sanjay Misra.
Analyzing the evolution of web services using fine-grained changes.
In *19th International Conference on Web Services (ICWS), 2012*, pages 392–399, 2012.

José Luis Ordiales Coscia, Marco Crasso, Cristian Mateos, Alejandro Zunino, and Sanjay Misra.
Predicting web service maintainability via object-oriented metrics: a statistics-based approach.
In *Computational Science and Its Applications–ICCSA 2012*, pages 19–39, 2012.

Ana Erika Camargo Cruz and Koichiro Ochimizu.
Towards logistic regression models for predicting fault-prone code across software projects.
In *Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on*, pages 460–463, 2009.

References XI

Melis Dagpinar and Jens H Jahnke.
Predicting maintainability with object-oriented metrics-an empirical comparison.
In *2013 20th Working Conference on Reverse Engineering (WCRE)*, pages 155–164, 2003.

Leandro N De Castro and Fernando J Von Zuben.
Learning and optimization using the clonal selection principle.
IEEE Transactions on Evolutionary Computation, 6(3):239–251, 2002.

Shyamala Doraisamy, Shahram Golzari, Noris Mohd, Md Nasir Sulaiman, and Nur Izura Udzir.
A study on feature selection and classification techniques for automatic genre classification of traditional malay music.
In *ISMIR*, pages 331–336, 2008.

Didier Dubois and Henri Prade.
Fuzzy real algebra: some results.
IEEE Transactions on Software Engineering, 2(4):327–348, 1979.

References XII

R. Slowinski (Ed).
Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory.
Kluwer Academic Publishers, Dordrecht, 1992.

Khaled El Emam, Saïda Benlarbi, Nishith Goel, and Shesh N. Rai.
The confounding effect of class size on the validity of object-oriented metrics.
IEEE Transactions on Software Engineering, 27(7):630–650, 2001.

Khaled El Emam, Walcelio Melo, and Javam C Machado.
The prediction of faulty classes using object-oriented design metrics.
56(1):63–75, 2001.

Ezgi Erturk and Ebru Akcapinar Sezer.
A comparison of some soft computing methods for software fault prediction.
Expert Systems with Applications, 42(4):1872–1879, 2015.

L. Etzkorn, J. Bansiya, and C. Davis.
Design and code complexity metrics for Object-Oriented classes.
Object-Oriented Programming, 12(10):35–40, 1999.

References XIII

Marios Fokaefs, Rimon Mikhaiel, Nikolaos Tsantalis, Eleni Stroulia, and Alex Lau.
An empirical study on web service evolution.
In *IEEE International Conference on Web Services (ICWS), 2011*, pages 49–56, 2011.

George Forman.
An extensive empirical study of feature selection metrics for text classification.
The Journal of machine learning research, 3:1289–1305, 2003.

Jerome H Friedman.
Multivariate adaptive regression splines.
The annals of statistics, 19:1–67, 1991.

Cesare Furlanello, Maria Serafini, Stefano Merler, and Giuseppe Jurman.
Entropy-based gene ranking without selection bias for the predictive classification of microarray data.
BMC bioinformatics, 4(1):1, 2003.

References XIV

M. Pezze G. Denaro and S. Morasca.
Towards industrially relevant fault-proneness models.
International Journal of Software Engineering and Knowledge Engineering, 13(4):395–417, 2003.

K. Gao and T. M. Khoshgoftaar.
A comprehensive empirical study of count models for software fault prediction.
IEEE Transactions for Reliability, 56(2):223–236, 2007.

Kehan Gao, Taghi Khoshgoftaar, and Amri Napolitano.
Exploring software quality classification with a wrapper-based feature ranking technique.
In *21st International Conference on Tools with Artificial Intelligence, 2009. ICTAI'09.*, pages 67–74, 2009.

Kehan Gao, Taghi M Khoshgoftaar, Huanjing Wang, and Naeem Seliya.
Choosing software metrics for defect prediction: an investigation on feature selection techniques.
Software: Practice and Experience, 41(5):579–606, 2011.

References XV

Narsimaih Gorla, Alan C. Benander, and Barbara A. Benander.
Debugging effort estimation using software metrics.
IEEE Transactions on Software Engineering, 16(2):223–231, February 1990.

Rinkaj Goyal, Pravin Chandra, and Yogesh Singh.
Suitability of knn regression in the development of interaction based software fault prediction models.
IERI Procedia, 6:15–21, 2014.

Tibor Gyimothy, Rudolf Ferenc, and Istvan Siket.
Empirical validation of object-oriented metrics on open source software for fault prediction.
31(10):897–910, 2005.

M.H. Halstead.
Elements of Software Sciencel.
Elsevier Science, New York, USA, 1977.

References XVI

Rachel Harrison, Steve J Counsell, and Reuben V Nithi.
An evaluation of the mood set of object-oriented software metrics.
IEEE Transactions on Software Engineering, 24(6):491–496, 1998.

John A Hartigan and Manchek A Wong.
Algorithm as 136: A k-means clustering algorithm.
Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1):100–108, 1979.

B. Henderson-Sellers.
Software Metrics.
Prentice-Hall, UK, 1996.

Martin Hitz and Behzad Montazeri.
Measuring coupling and cohesion in object-oriented systems.
In *Proceedings of the International Symposium on Applied Corporate Computing*, volume 50, pages 75–76. Seattle.

References XVII

Allen E.B Hudepohl J.P Hochman R, Khoshgoftar T.M.
Evolutionary neural networks: a robust approach to software reliability problems.
In *Proceedings of the Eight International Symposium on Software Reliability Engineering*, pages 13–26, 1997.

Ross Huitt and Norman Wilde.
Maintenance support for object-oriented programs.
IEEE Transactions on Software Engineering, 18(12):1038–1044, 1992.

Rob J Hyndman and Anne B Koehler.
Another look at measures of forecast accuracy.
International Journal of Forecasting, 22(4):679–688, 2006.

A. Idri, A. Abran, and S. Mbarki.
An experiment on the design of radial basis function neural networks for software cost estimation.
In *2nd IEEE International Conference on Information and Communication Technologies*, volume 1, pages 230–235, 2006.

References XVIII

Y Kumar Jain and Santosh Kumar Bhandare.
Min max normalization based data perturbation method for privacy protection.
International Journal of Computer and Communication Technology, 2(8):45–50, 2011.

Cukic B Jiang Y and Ma Y.
Techniques for evaluating fault prediction models.
Empirical Software Engineering, 13(5):561–595, 2008.

Cong Jin and Jin-An Liu.
Applications of support vector mathine and unsupervised learning for predicting maintainability using object-oriented metrics.
In *Second International Conference on Multimedia and Information Technology (MMIT), 2010*, pages 24–27, 2010.

Ho-Won Jung, Seung-Gweon Kim, and Chang-Shin Chung.
Measuring software product quality: A survey of iso/iec 9126.
IEEE software, 21(5):88–92, 2004.

References XIX

Satyen Kale, Ravi Kumar, and Sergei Vassilvitskii.
Cross-validation and mean-square stability.
In *Innovations in Computer Science*, pages 487–495, 2011.

B. K. Kang and J. M. Bieman.
Cohesion and reuse in an Object-Oriented system.
In *Proceedings of the ACM SIGSOFT Symposium on software reuseability*, pages 259–262. Seattle, March 1995.

S Kanmani, V Rhymend Uthariaraj, V Sankaranarayanan, and P Thambidurai.
Object-oriented software fault prediction using neural networks.
Information and software Technology, 49(5):483–492, 2007.

Heena Kapila and Satwinder Singh.
Analysis of ck metrics to predict software fault-proneness using bayesian inference.
International Journal of Computer Applications, 74(2), 2013.

References XX

Arvinder Kaur, Kamaldeep Kaur, and Ruchika Malhotra.
Soft computing approaches for prediction of software maintenance effort.
Information and Software Technology, 1(16):975–987, 2010.

Jaswinder Kaur, Satwinder Singh, Karanjeet Singh Kahlon, and Pourush Bassi.
Neural network-a novel technique for software effort estimation.
International Journal of Computer Theory and Engineering, 2(1):17–19, 2010.

Hudepohl J.P Aud S.J Khoshgoftaaar T.M, Allen E.B.
Application of neural networks to software quality modeling of a very large telecommunications systems.
IEEE Transactions on Neural Networks, 8(4):902–909, 1997.

T. M Khoshgoftaar, E. Geleyn, and K. GAo.
An empirical study of the impact of count models predictions on module-order models.
In *Proceedings of 8th International Symposium on Software metrics*, pages 161–172, 2002.

References XXI

T. M Khoshgoftaar, N. Seliya, and N. Sundaresh.
An empirical study of predicting software faults with case based reasoning.
Software Quality Journal, 14(2):85–111, 2006.

Kim and Ji-Hyun.
Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap.
Computational Statistics and Data Analysis, 53(11):3735–3745, 2009.

Kenji Kira and Larry A Rendell.
A practical approach to feature selection.
In *Proceedings of the ninth international workshop on Machine learning*, pages 249–256, 1992.

R. Kohavi.
A study of cross-validation and bootstrap for accuracy estimation and model selection.
In *Proceedings of the 14th International Joint Conference on Artificial Intelligence, San Mateo*, pages 1137–1143, 1995.

References XXII

R. Kohavi.
Relation between software metrics and maintainability.
In *Proceedings of the FESMA99 International Conference, Federation of European Software Measurement Associations, Amsterdam, The Netherlands*, pages 465–476, 1999.

Ron Kohavi and George H John.
Wrappers for feature subset selection.
Artificial intelligence, 97(1):273–324, 1997.

P. Devanbu L. Briand and W. Melo.
An investigation into coupling measures for c++.
In *Proceedings of International Conference on Software Engineering Association for Computing Machinery*, pages 412–421, 1997.

Al Lake and Curtis Cook.
Use of factor analysis to develop oop software complexity metrics.
In *Proceedings of 6th Annual Oregon Workshop on Software Metrics, Silver Falls, Oregon*, 1994.

References XXIII

YS Lee, BS Liang, SF Wu, and FJ Wang.
Measuring the coupling and cohesion of an object-oriented program based on information flow.
In *Proceedings of International Conference on Software Quality, Maribor, Slovenia*, pages 81–90, 1995.

K. Levenberg.
A method for the solution of certain non-linear problems in least squares.
Quarterly of Applied Mathematics, 2(2):164–168, 1944.

W. Li and S. Henry.
Maintenance metrics for the Object-Oriented paradigm.
In *Proceedings of First International Software Metrics Symposium*, pages 52–60, 1993.

M. Lorenz and J. Kidd.
Object-Oriented Software Metrics.
Prentice-Hall, NJ, Englewood, 1994.

References XXIV

Hongmin Lu, Yuming Zhou, Baowen Xu, Hareton Leung, and Lin Chen.
The ability of object-oriented metrics to predict change-proneness: a meta-analysis.
Empirical software engineering, 17(3):200–242, 2012.

Ruchika Malhotra and Anuradha Chug.
Application of group method of data handling model for software maintainability prediction using object oriented systems.
International Journal of System Assurance Engineering and Management, 5(2):165–173, 2014.

Ruchika Malhotra and Ankita Jain.
Fault prediction using statistical and machine learning methods for improving software quality.
Journal of Information Processing Systems, 8(2):241–262, 2012.

References XXV

Ruchika Malhotra and Yogesh Singh.
On the applicability of machine learning techniques for object oriented software fault prediction.
Software Engineering: An International Journal, 1(1):24–37, 2011.

D. W. Marquardt.
An algorithm for the least-squares estimation of nonlinear parameters.
SIAM Journal of Applied Mathematics, 11(2):431–441, 1963.

R. Martin.
Object-oriented design quality metrics an analysis of dependencies.
In *Proc. Workshop Pragmatic and Theoretical Directions in Object-Oriented Software Metrics*, October 1994.

Cristian Mateos, Marco Crasso, Alejandro Zunino, and Jos? Luis Ordiales Coscia.
Detecting wsdl bad practices in code–first web services.
International Journal of Web and Grid Services, 7(4):357–387, 2011.

References XXVI

Thomas J. McCabe.
A complexity measure.
IEEE Transactions on Software Engineering, 2(4):308–320, December 1976.

Andrew McCallum and Kamal Nigam.
A comparison of event models for naive bayes text classification.
In *AAAI-98 workshop on learning for text categorization*, pages 42–48, 1998.

Warren McCulloch and Walter Pitts.
A logical calculus of ideas immanent in nervous activity.
Bulletin of Mathematical Biophysics, 5(4):115–133, 1943.

W. Melo and F. B. E. Abreu.
Evaluating the impact of Object-Oriented design on software quality.
In *Proceedings of the 3rd International Software Metrics Symposium*, pages 90–99, 1996.

Tim Menzies, Bora Caglayan, Zhimin He, Ekrem Kocaguneli, Joe Krall, Fayola Peters, and Burak Turhan.
The promise repository of empirical software engineering data, June 2012.

References XXVII

Tim Menzies, Zhihao Chen, Jairus Hihn, and Karen Lum.
Selecting best practices for effort estimation.
IEEE Transactions on Software Engineering, 32(11):883–895, 2006.

Bharavi Mishra, K Shukla, et al.
Defect prediction for object oriented software using support vector based fuzzy classification model.
International Journal of Computer Applications, 60(15), 2012.

Subhas Chandra Misra.
Modeling design/coding factors that drive maintainability of software systems.
Software Quality Journal, 13(3):297–320, 2005.

Naouel Moha, Francis Palma, Mathieu Nayrolles, Benjamin Joyen Conseil, Yann-Gaël Guéhéneuc, Benoit Baudry, and Jean-Marc Jézéquel.
Specification and detection of soa antipatterns.
In *Service-Oriented Computing*, pages 1–16, 2012.

References XXVIII

J Moody and J Darken C.
Fast learning in networks of locally-tunes processing units.
Neural Computation, 1(2):284–294, 1989.

Nachiappan Nagappan, Laurie Williams, Mladen Vouk, and Jason Osborne.
Early estimation of software quality using in-process testing metrics: a controlled case study.
In *ACM SIGSOFT Software Engineering Notes*, volume 30, pages 1–7, 2005.

Jasmina Novakovic.
The impact of feature selection on the accuracy of naïve bayes classifier.
In *18th Telecommunications forum TELFOR*, pages 1113–1116, 2010.

Hector M Olague, Letha H Etzkorn, Sampson Gholston, and Stephen Quattlebaum.
Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes.
Software Engineering, IEEE Transactions on.

References XXIX

Hector M Olague, Letha H Etzkorn, Sampson Gholston, and Stephen Quattlebaum.
Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes.
IEEE Transactions on Software Engineering, 33(6):402–419, 2007.

Paul Oman and Jack Hagemeister.
Construction and testing of polynomials predicting software maintainability.
Journal of Systems and Software, 24(3):251–266, 1994.

Ganesh J Pai and Joanne Bechta Dugan.
Empirical analysis of software fault content and fault proneness using bayesian methods.
IEEE Transactions on Software Engineering, 33(10):675–686, 2007.

H Pao, Y.
Adaptive Pattern Recognition and neural networks. Reading.
MA addison, Wesley, 1989.

References XXX

Z. Pawlak.
Rough sets.
International Journal of Computer and Information Sciences, 11(5):341–356, 1982.

Mikhail Perepletchikov, Caspar Ryan, and Keith Frampton.
Cohesion metrics for predicting maintainability of service-oriented software.
In *Seventh International Conference on Quality Software (QSIC 2007)*, pages 328–335. IEEE, 2007.

Mikhail Perepletchikov, Caspar Ryan, Keith Frampton, and Zahir Tari.
Coupling metrics for predicting maintainability in service-oriented designs.
In *Software Engineering Conference, 2007. ASWEC 2007. 18th Australian*, pages 329–340. IEEE, 2007.

Mikhail Perepletchikov, Caspar Ryan, and Zahir Tari.
The impact of service cohesion on the analyzability of service-oriented software.
IEEE Transactions on Services Computing, 3(2):89–103, 2010.

References XXXI

Robin L Plackett.
Karl pearson and the chi-squared test.
International Statistical Review/Revue Internationale de Statistique, 51(1):59–72, 1983.

N Narayanan Prasanth, S Ganesh, and GA Dalton.
Prediction of maintainability using software complexity analysis: An extended frt.
In *International Conference on Computing, Communication and Networking, 2008. ICCCn 2008*, pages 1–9, 2008.

N Narayanan Prasanth, S Ganesh, and GA Dalton.
A quantitative approach to software maintainability prediction.
In *International Conference on Information Technology and Applications (IFITA), 2010*, pages 105–108, 2010.

Mehwish Riaz, Emilia Mendes, and Ewan Tempero.
Predicting maintenance effort with function points.
In *International Conference on Software Maintenance, 1997.*, pages 32–39, 1997.

References XXXII

Mehwish Riaz, Emilia Mendes, and Ewan Tempero.
A systematic review of software maintainability prediction and metrics.
In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, pages 367–377, 2009.

Wagner S.
A literature survey of the quality economics of defect-detection techniques.
In *Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering (ISESE)*, pages 194–203, 2006.

Scott L Schneberger.
Distributed computing environments: effects on software maintenance difficulty.
Journal of Systems and Software, 37(2):101–116, 1997.

Y. Shan, R.I McKay, C.J Lokan, D.L Essam, and Y.Chang.
Software project effort estimation using genetic programming.
In *Proceedings of IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions*, pages 1108–1112, July 2002.

References XXXIII

Raed Shatnawi and Wei Li.
The effectiveness of software metrics in identifying error-prone classes in post-release software evolution process.
Journal of systems and software, 81(11):1868–1882, 2008.

Bingu Shim, Siho Choue, Suntae Kim, and Sooyong Park.
A design quality model for service-oriented architecture.
In *2008 15th Asia-Pacific Software Engineering Conference*, pages 403–410. IEEE, 2008.

Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra.
Software fault proneness prediction using support vector machines.
In *Proceedings of the world congress on engineering*, volume 1, pages 1–3, 2009.

Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra.
Empirical validation of object-oriented metrics for predicting fault proneness models.
Software quality journal, 18(1):3–35, 2010.

References XXXIV

K. Srinivasan and D. Fisher.
Machine learning approaches to estimating software development effort.
IEEE Transactions on Software Engineering, 21(2):126–137, February 1995.

Ramanath Subramanyam and Mayuram S. Krishnan.
Empirical analysis of ck metrics for object-oriented design complexity: Implications for software defects.
IEEE Transactions on Software Engineering, 29(4):297–310, 2003.

Johan AK Suykens, Jos De Brabanter, Lukas Lukas, and Joos Vandewalle.
Weighted least squares support vector machines: robustness and sparse approximation.
Neurocomputing.

Mende T and Koschke R.
Revisiting the evaluation of defect prediction models.
In *Proceedings of the 5th International Conference on Predictor Models in Software Engineering (PROMISE)*, pages 1–10, 2009.

References XXXV

Mende T and Koschke R.
Effort-aware defect prediction models.
In *Proceedings of 14th IEEE European Conference on Software Maintenance and Re-engineering (CSMR)*, pages 107–116, 2010.

R. Ferenc T. Gyimothy and I. Siket.
Empirical validation of Object-Oriented metrics on open source software for fault prediction.
IEEE Transactions on Software Engineering, 31(10):897–910, October 2005.

Mei-Huei Tang, Ming-Hung Kao, and Mei-Hwa Chen.
An empirical study on object-oriented metrics.
In *Software Metrics Symposium, 1999. Proceedings. Sixth International*, pages 242–249. IEEE, 1999.

D. P. Tegarden, S. D. Sheetz, and D. E. Monarchi.
A software complexity model of Object-Oriented systems.
Decision Support Systems, 13(3):241–262, 1995.

References XXXVI

Piotr Tomaszewski, Jim Håkansson, Håkan Grahn, and Lars Lundberg.
Statistical models vs. expert estimation for fault prediction in modified code–an industrial case study.
Journal of Systems and Software, 80(8):1227–1238, 2007.

Chikako Van Koten and AR Gray.
An application of bayesian network for predicting object-oriented software maintainability.
Journal of Materials Processing Technology, 48(1):59–67, 2006.

D Wang and JA Romagnoli.
Robust multi-scale principal components analysis with applications to process monitoring.
Journal of Process Control, 15(8):869–882, 2005.

G. Witting and G. Finnie.
Estimating software development effort with connectionist models.
In *Proceedings of the Information and Software Technology Conference*, pages 469–476, 1997.

References XXXVII

W.Li and S.Henry.
Object-oriented metrics that predicts maintainablity.
Journel of System and Software, 32(2):111–122, 1993.

Fangjun Wu.
Empirical validation of object-oriented metrics on nasa for fault prediction.
In *Advances in Information Technology and Education*, pages 168–175. 2011.

Yijun Yu, Jianguo Lu, Juan Fernandez-Ramil, and Phil Yuan.
Comparing web services with other software components.
In *Web Services, 2007. ICWS 2007. IEEE International Conference on*, pages 388–397. IEEE, 2007.

Jialin Zhou, Zhengcheng Duan, Yong Li, Jianchun Deng, and Daoyuan Yu.
Pso-based neural network optimization and its utilization in a boring machine.
Journal of Materials Processing Technology, 178(1):19–23, 2006.

References XXXVIII

Yuming Zhou and Hareton Leung.
Empirical analysis of object-oriented design metrics for predicting high and low severity faults.
32(10):771–789, 2006.

Yuming Zhou and Hareton Leung.
Predicting object-oriented software maintainability using multivariate adaptive regression splines.
Journal of Materials Processing Technology, 80(8):1349–1361, 2007.

Yuming Zhou and Baowen Xu.
Predicting the maintainability of open source software using design metrics.
Wuhan University Journal of Natural Sciences, 13(1):14–20, 2008.

Yuming Zhou, Baowen Xu, and Hareton Leung.
On the ability of complexity metrics to predict fault-prone classes in object-oriented systems.
Journal of Systems and Software, 83(4):660–674, 2010.