Indicators of Structural Stability of Object-Oriented Designs: A Case Study

Mahmoud O. Elish and David Rine
Department of Computer Science
George Mason University
Fairfax, VA 22030, USA
{melish,drine}@gmu.edu

Abstract

The structural stability of an object-oriented design (OOD) refers to the extent to which the structure of the design is preserved throughout the evolution of the software from one release to the next. This paper empirically investigates potential indicators of measures of structural stability of OODs. Both product-related and process-related indicators are considered. These indicators were evaluated through a case study that involves 13 successive releases of Apache Ant. The results showed that each one of the stability metrics is significantly correlated with at least one of the investigated indicators. To make early predictions of the values of each one of the stability metrics, statistically significant regression models were constructed from subsets of the investigated indicators.

Keywords: *Structural stability, software metrics, software evolution, object-oriented designs.*

1. Introduction

Considerable resources and effort are invested and spent in deriving a high quality software design. Such design, presumable correct, represents an important organizational *asset* whose long-term value should be enforced and preserved. Fundamental concepts of design structure quality have traditionally included modularity, abstraction, and encapsulation. This paper focuses primarily on modularity qualities. The structural stability of an object-oriented design (OOD) refers to the extent to which the structure of the design is preserved, that is remains invariant, throughout the evolution of the software from one release to the next. Structural stability of a design is a sign of its capability to evolve while preserving its structure. The structure of an OOD, in this paper, is defined by the design classes and the relationships as they are the two most fundamental building blocks. Structural design takes the form of a directed graph whose vertices represent classes and whose edges represent relationships between these classes.

The availability of adequate indicators of design structural stability can give software managers early insight into structural stability of designs and trends in software evolution. This enables them to effectively assess, manage and control the design structural stability of long-lived software systems. According to DeMarco's principle [8], "you cannot control what you cannot measure."

None of the pervious research studies focused on assessing the structural stability of OODs through software releases nor investigated indicators of the structural stability of OODs. Jazayeri [13] applied retrospective analyses to successive releases of a large telecommunication software system (not an object-oriented software) to evaluate its architectural stability with simple size measures, coupling measures, and color visualization to observe phenomena about the evolution of the software across releases. Bansiya [5] proposed a methodology to assess the stability of framework architectures (limited to class inheritance hierarchies) over successive releases by determining the extent-of-change between releases. Mattsson and Bosch [14, 15] extended Bansiya's method with an additional aggregated metric, the relative-extent-of-change metric. Grosser et al. [10, 11] proposed a case-based reasoning approach for predicting stability of Java classes.

The objective of this paper is to empirically investigate potential product-related and process-related indicators of measures of structural stability of OODs. In other words, the paper aims to investigate and test assertions about significant correlations between measures of the investigated indicators and measures of OOD structural stability metrics through a case study of 13 successive releases of Apache Ant.

In Proceedings of the 29th Annual NASA/IEEE Software Engineering Workshop (SEW 29), pp. 183- 192, Maryland, USA, Apr. 2005.

The rest of this paper is organized as follows. Section 2 defines a suite of metrics for measuring OOD structural stability. Section 3 defines the investigated product-related and process-related stability indicators. Section 4 reports the case study and its results. Section 5 concludes the paper.

2. Measuring OOD structural stability

OOD structural stability, in this paper, is measured through a suite of metrics (thereafter *stability metrics suite*) that evaluates the extent to which the structure of an OOD remained stable from one software release to the next throughout the software evolution. The stability of a design structure is a two-dimension concept. The first dimension considers the size of design structural modifications from one software release to the next to determine *how much* of the original design structure remained invariant, i.e. stable. The second dimension considers the period of time from one release to the next to determine *how long* the structure remained invariant, i.e. stable. In order to assess both dimensions of design structural stability, the stability metrics suite includes size-based metrics and time-based metrics.

2.1. Size-based metrics

Size-based metrics of design structural stability calculate the size of the changes made to a design structure as well as the size of the design structure part that remained stable from one software release to the next. These metrics are measured by comparing every two successive software releases throughout the software evolution.

Since the structure of an OOD is defined by the design classes and the relationships, the extent to which these two fundamental building blocks (classes and relationships) are preserved from one release to the next needs to be measured by the size-based stability metrics. Therefore, two categories of size-based metrics of OOD structural stability are considered: class-based metrics and relationship-based metrics.

2.1.1. Class-based metrics. Class-based metrics of OOD structural stability calculate the extent to which design classes are preserved from one software release to the next. As software undergoes changes, existing classes may be modified or deleted, and new classes may be added.

Four class-based stability metrics are included in the stability metrics suite: *NSC*, *NAC*, *NDC*, and *NMC*. The *NSC* metric counts the number of classes that remained stable (i.e. were not added, deleted, or modified) between any two successive software releases. *NAC*, *NDC*, and *NMC* metrics count the number of classes that were added, deleted, and modified respectively between any two successive software releases.

The three-letter acronyms of class-based stability metrics are interpreted as follows. The first letter is *N*, which stands for "number of." The second letter represents the kind of change, i.e. *S* for stable, *A* for added, *D* for deleted, and *M* for modified. The last letter is *C*, which stands for 'classes'.

2.1.2. Relationship-based metrics. Relationship-based metrics of OOD structural stability calculate the extent to which relationships between classes in a design are preserved from one software release to the next. As software undergoes changes, existing class relationships may be deleted and new relationships may be added. These include relationships of all kinds, i.e. generalization, aggregation, dependency, and association. A *generalization relationship* is a relationship between a more general class (superclass) and more specific class (subclass). An *aggregation relationship* exists between two classes if one is part of the other, i.e. if one is the type of an attribute of the other. A *dependency relationship* exists between two classes if one is the return type of a method of the other or the type of a parameter of a method of the other. An *association relationship* exists between two classes if one invokes one or more methods of the other and/or references one or more attributes of the other.

Twelve relationship-based stability metrics are included in the stability metrics suite; three for each kind of relationships. *NSGR*, *NAGR*, and *NDGR* metrics count the number generalization relationships that were stable (neither added nor deleted), added, and deleted respectively between any two successive software releases. *NSAR*, *NAAR*, and *NDAR* metrics count the number aggregation relationships that were stable, added, and deleted respectively between any two successive software releases. *NSDR*, *NADR*, and *NDDR* metrics count the number dependency relationships that were stable, added, and deleted respectively between any two successive software releases. Finally, *NSSR*, *NASR*, and *NDSR* metrics count the number association relationships that were stable, added, and deleted respectively between any two successive software releases.

The four-letter acronyms of relationship-based stability metrics are interpreted as follows. The first letter is *N*, which stands for "number of." The second letter represents the kind of change, i.e. *S* for stable, *A* for added, and *D* for deleted. The last two letters

represent the kind of relationship, i.e. *GR* for generalization relationships, *AR* for aggregation relationships, *DR* for dependency relationships, and *SR* for association relationships.

2.2. Time-based metric

The size-based metrics defined in the previous section capture one dimension of design structural stability, i.e. *how much* of the design structure (classes and relationships) remained stable between two successive software releases. Another important dimension is *how long* the design structure remained stable, which is measured by time-based metrics. Structural stability of designs cannot be adequately assessed unless these two dimensions are considered. If two designs have relatively the same size of structural modifications since their first release, but one is 3 years old and the other is 10 years old, the older would intuitively be considered more stable.

The stability metrics suite includes one time-based stability metric, which is *time between releases* (*TBR*) to quantify the *how long* dimension of design structural stability. The *TBR* metric can be measured in days, weeks, months, etc. In this paper, it is measured in days.

3. Potential indicators of OOD structural stability

This section discusses potential indicators of OOD structural stability. Indicators of design structural stability can be classified into two groups: product-related and process-related indicators. Product-related indicators are measures of design structural characteristics that are collected from the design version at release *i* and then used to predict design structural stability from release *i* to release *i+1* (following release). Process-related indicators are measures that are collected from maintenance related activities performed on release *i* and then used to predict design structural stability from release *i* to release *i+1*.

3.1. Product-related indicators

This paper adapts two suites of metrics for OODs as potential product-related indicators of the structural stability of OODs: the Chidamber and Kemerer (C&K) metrics suite [7] and the MOOD metrics suite [4, 12]. These two suites of metrics were selected because of their wide acceptance among the software engineering community. They have been used by numerous previous empirical studies, and perhaps have been cited the most. In addition, they capture important dimensions of OOD structural characteristics: size/complexity, inheritance, cohesion, and coupling, encapsulation, and polymorphism. The goal is to assess the metrics of these two suites as indicators (early predictors) of OOD structural stability.

3.1.1. C&K metrics suite. The C&K metrics suite consists of six metrics: weighted methods per class (*WMC*), depth of inheritance tree (*DIT*), number of children (*NOC*), coupling between object classes (*CBO*), response for a class (*RFC*), and lack of cohesion in methods (*LCOM*). Since these metrics are class-level metrics, the average of each individual metric is investigated as design-level stability indicator. The adopted metrics are defined next.

- *Average Weighted Methods per Class (AWMC):* The *AWMC* metric is defined as the average of the sum of the cyclomatic complexities of all methods in a class.

- *Average Depth of Inheritance Tree per Class (ADIT):* The *ADIT* metric is defined as the average depth of a class within its inheritance hierarchy.

- *Average Number of Children per Class (ANOC):* The *ANOC* metric is defined as the average number of direct subclasses of a class.

- *Average Coupling between Object Classes (ACBO):* The *ACBO* metric is defined as the average number of classes to which a class is coupled. A class is coupled to all classes with which it has any kind of relationship.

- *Average Response for a Class (ARFC):* The *ARFC* metric is defined as the average number of methods in the set of all methods that can be invoked in response to a message sent to an object of a class.

- *Average Lack of Cohesion in Methods per Class (ALCOM):* The *ALCOM* metric is defined as the average of [the number of method pairs whose similarity is zero (P) minus the number of method pairs whose similarity is not zero (Q) in a class, if P > Q. Otherwise *LCOM* of a class is zero]. The similarity of two methods is computed by counting the number of instance variables (attributes) that are used by both of them.

3.1.2. MOOD metrics suite. The MOOD metrics suite consists of six metrics: method hiding factor (*MHF*), attribute hiding factor (*AHF*), method inheritance factor (*MIF*), attribute inheritance factor (*AIF*), coupling factor (*CF*), and polymorphism factor (*PF*). These metrics are informally defined next, while their formal definition and details are in [4, 12].

- *Method Hiding Factor (MHF):* The *MHF* is defined as the ratio of the sum of the invisibilities of all methods defined in all classes to the total number of methods in a design.

- *Attribute Hiding Factor (AHF):* The *AHF* is defined as the ratio of the sum of the invisibilities of all attributes defined in all classes to the total number of class attributes in a design.

- *Method Inheritance Factor (MIF):* The *MIF* metric is defined as the ratio of the number of inherited (and not overridden) methods in all classes to the total number of available methods (locally defined plus inherited) for all classes in a design.

- *Attribute Inheritance Factor (AIF):* The *AIF* metric is defined as the ratio of the number of inherited attributes in all classes to the total number of available attributes (locally defined plus inherited) for all classes in a design.

- *Coupling Factor (CF):* The *CF* metric is defined as the ratio of the number of class couplings to the maximum possible number of class couplings in a design.

- *Polymorphism Factor (PF):* The *PF* metric is defined as the ratio the number of methods that redefine inherited methods to the maximum number of possible distinct polymorphic situations.

3.2. Process-related indicators

This paper investigates software age and change requests as potential process-related indicators of the structural stability of OODs. These indicators are defined next.

3.2.1. Software age. Software age (*AGE*) at a particular release time is a measure of the number of days elapsed since its first release.

3.2.2. Change requests. As software moves from one release to the next, it resolves a number of change requests. Each change request can be corrective, perfective, or adaptive request. The numbers of each kind of change requests resolved between two successive releases are investigated as potential indicators of the design structural stability between these two successive releases. Precisely, this paper investigates number of corrective change requests (*NCCR*), number of adaptive change requests (*NACR*), and number of perfective change requests (*NPCR*) that were resolved between two successive releases.

4. Case study

The potential indicators of OOD structural stability described in Section 3 were empirically evaluated through a case study of 13 successive releases of Apache Ant [1], which is an open source Java-based build tool. Apache Ant software project is more than four years old. Its most recent release (1.6.2) has more than 200K lines of code, and more than 1200 classes. In this paper, 13 successive releases of Apache Ant were analyzed, from its first release (1.1) to its most recent release (1.6.2). Precisely, the releases are 1.1, 1.2, 1.3, 1.4, 1.4.1, 1.5, 1.5.1, 1.5.2, 1.5.3, 1.5.4, 1.6, 1.6.1, and 1.6.2.

Using the Goal/Question/Metric (GQM) template [6] for goal definition, the goal of this case study is defined as follows: *Analyze* the potential OOD structural stability indicators *for the purpose of* empirical evaluation *with respect to* the capability of being used as predictors of measures of the metrics of the stability metrics suite *from the point of view of* software engineering practitioners and researchers *in the context of* Apache Ant.

4.1. Dependant variables

The dependant variables are the metrics of the stability metrics suite, which are described in Section 2. These metrics are *NSC, NAC, NDC, NMC*, *NSGR, NAGR, NDGR, NSAR, NAAR, NDAR, NSDR, NADR, NDDR, NSSR, NASR, NDSR*, and *TBR*.

4.2. Independent variables

The independent variables are the potential product-related and process-related indicators of OOD structural stability, which are described in Section 3. There are 12 product-related indicators (*AWMC*, *ADIT*, *ANOC*, *ACBO*, *ARFC*, *ALCOM*, *MHF*, *AHF*, *MIF*, *AIF*, *CF*, and *PF*), and 4 process-related indicators (*AGE*, *NCCR*, *NACR*, and *NPCR*).

4.3. Hypothesis

The general hypothesis that is tested by this case study is that there is a significant correlation between each of the potential stability indicators (independent variables) that are collected from release *i* and each of the metrics of the stability metrics suite (dependant variables) from release *i* to release *i+1*, and thus these indicators can be used as inputs to stability prediction models.

4.4. Data collection

The dependant variables were measured as follows. The *ExamDiff Pro* file comparison tool [2] was used to calculate the class-based stability metrics from each release to the next by comparing classes between releases. Comment and blank lines were excluded in class comparison. A class was considered modified if at least one of its lines of code was deleted or changed, or at lease one new line of code was added to it. A prototype metrics tool that has been developed as part of this research was used to calculate the relationship-based stability metrics from each release to the next by comparing relationships between releases. Details about this prototype tool are left for another paper. The *TBR* metric was simply measured between any two successive releases using their release dates.

The independent variables were measured as follows. All product-related indicators were measured using the developed prototype metrics tool, except the *AWMC*, *ARFC*, and *ALCOM* metrics that were measured using the *JStyle* metrics tool [3]. *NCCR*, *NACR*, and *NPCR* were collected from the release note of each release of Apache Ant that lists corrective, adaptive, and perfective change requests that were resolved since the previous release. The *AGE* metric of each release was simply measured by counting the number of days elapsed since the first release.

4.5. Results

The results of the empirical evaluation of the potential indicators of OOD structural stability in this case study are reported and analyzed in this section.

4.5.1. Descriptive statistics. Table 1 provides descriptive statistics for the measures of the 17 stability metrics of the stability metrics suite that were collected from the releases of Apache Ant. In general, class relationships were more stable than classes over the releases of Apache Ant as the ratio of stable relationships to unstable relationships was more than the ratio of stable classes to unstable classes. Deletions of classes and relationships were relatively low compared to additions and modifications. Measures of the *TBR* metric vary from 37 days to 272 days with an average of ~122 days between releases.

Table 1. Descriptive statistics for measures of the 17 stability metrics

Metric	Minimum	Maximum	Mean	Std. Dev.
NSC	23	920	459.92	340.10
NAC	0	328	109.58	125.32
NDC	0	229	39.42	85.14
NMC	8	466	122.17	142.47
NSGR	52	790	452.58	250.18
NAGR	2	312	82.58	105.21
NDGR	0	100	19.25	30.09
NSAR	34	387	235.92	113.23
NAAR	0	115	40.42	47.01
NDAR	0	61	9.58	18.97
NSDR	318	1239	809.67	278.75
NADR	0	324	98.58	115.74
NDDR	0	136	19.08	40.36
NSSR	214	3128	1762.58	953.26
NASR	2	984	316.25	379.18
NDSR	0	421	67.75	123.93
TBR	37	272	121.58	66.79

Table 2 provides descriptive statistics for the measures of the 16 investigated potential stability indicators that were collected from Apache Ant releases. Over all the releases, it was found that 48% of the changes were perfective maintenance; 40% were corrective; and 12% were adaptive.

Table 2. Descriptive statistics for measures of the 16 potential stability indicators

Indicator	Minimum	Maximum	Mean	Std. Dev.
AWMC	12.02	16.18	14.28	1.02
ADIT	1.70	2.02	1.93	0.09
ANOC	0.85	1.04	0.98	0.07
ACBO	12.37	13.02	12.54	0.19
ARFC	19.06	23.34	20.73	1.00
ALCOM	31.14	47.16	41.03	3.93
MHF	0.0386	0.0753	0.0685	0.0131
AHF	0.8333	0.9014	0.8669	0.0156
MIF	0.4971	0.6434	0.5851	0.0472
AIF	0.2623	0.3187	0.3034	0.0183
CF	0.0031	0.0252	0.0072	0.0066
PF	0.0380	0.075	0.0508	0.0114
AGE	0	1304	695.08	450.08
NCCR	4	62	24.50	19.42
NACR	0	22	7.50	8.59
NPCR	1	118	29.50	37.10

4.5.2. Univariate analysis. Correlation analysis was performed, at a 0.05 level of significance (95% confidence level), to test for existence of significance correlation between measures of each individual potential stability indicator (independent variable) and measures of each individual stability metric (dependant

Table 3. Correlation coefficients: stability metrics vs. potential stability indicators

	AWMC	ADIT	ANOC	ACBO	ARFC	ALCOM	MHF	AHF	MIF	AIF	CF	PF	AGE	NCCR	NACR	NPCR
NSC	-0.17	**0.73**	**0.83**	-0.48	-0.26	0.31	**0.59**	0.16	**0.78**	**0.64**	**-0.66**	**-0.80**	**0.81**	-0.34	**-0.74**	-0.48
NAC	0.18	**-0.63**	-0.36	0.55	0.36	-0.31	-0.34	-0.29	-0.54	-0.08	0.30	0.48	-0.47	0.55	0.57	0.51
NDC	**-0.58**	0.03	-0.15	-0.07	-0.49	**-0.59**	0.11	-0.47	-0.38	-0.18	-0.11	0.40	-0.13	-0.07	0.23	0.01
NMC	-0.13	0.00	-0.03	0.08	-0.09	-0.15	0.16	-0.16	-0.07	-0.02	-0.12	0.04	0.09	**0.92**	**0.74**	**0.96**
NSGR	-0.22	**0.83**	**0.94**	-0.50	-0.30	0.33	**0.71**	0.12	**0.87**	**0.72**	**-0.79**	**-0.91**	**0.97**	0.07	-0.50	-0.09
NAGR	0.03	-0.42	-0.39	0.47	0.19	-0.29	-0.17	-0.30	-0.46	-0.24	0.19	0.43	-0.31	**0.73**	**0.83**	**0.84**
NDGR	**-0.66**	-0.06	-0.39	0.14	-0.47	**-0.73**	0.11	**-0.69**	**-0.59**	-0.39	-0.06	**0.63**	-0.29	0.53	**0.78**	0.56
NSAR	-0.24	**0.82**	**0.92**	-0.43	-0.29	0.28	**0.76**	0.05	**0.83**	**0.71**	**-0.83**	**-0.89**	**0.97**	0.09	-0.50	-0.08
NAAR	0.02	-0.53	-0.50	0.47	0.18	-0.36	-0.34	-0.32	**-0.58**	-0.30	0.33	**0.58**	-0.44	**0.67**	**0.87**	**0.81**
NDAR	**-0.68**	-0.09	-0.40	0.13	-0.49	**-0.78**	0.11	**-0.69**	**-0.63**	-0.40	-0.05	**0.66**	-0.33	0.51	**0.77**	0.53
NSDR	-0.21	**0.78**	**0.91**	-0.38	-0.23	0.29	**0.69**	0.03	**0.82**	**0.73**	**-0.80**	**-0.87**	**0.97**	0.11	-0.46	-0.05
NADR	0.27	-0.46	-0.48	0.22	0.25	0.00	-0.47	0.10	-0.33	-0.35	0.51	0.39	-0.38	**0.59**	**0.81**	**0.78**
NDDR	-0.47	-0.38	-0.36	0.33	-0.21	**-0.80**	-0.19	**-0.72**	**-0.70**	-0.16	0.11	**0.75**	-0.43	0.27	0.50	0.30
NSSR	-0.21	**0.83**	**0.92**	-0.46	-0.28	0.33	**0.71**	0.10	**0.86**	**0.71**	**-0.79**	**-0.90**	**0.97**	0.08	-0.49	-0.08
NASR	0.03	-0.45	-0.40	0.43	0.17	-0.31	-0.23	-0.29	-0.48	-0.22	0.24	0.46	-0.34	**0.74**	**0.84**	**0.86**
NDSR	**-0.65**	-0.14	-0.43	0.13	-0.47	**-0.78**	0.03	**-0.68**	**-0.65**	-0.39	0.01	**0.70**	-0.37	0.49	**0.77**	0.52
TBR	-0.21	-0.15	-0.22	0.21	-0.08	-0.35	0.03	-0.35	-0.36	-0.19	-0.01	0.32	-0.19	**0.70**	**0.71**	**0.60**

variable) under investigation. Table 3 presents the correlation coefficients, where significant values are in boldface.

The following observations can be made from Table 3. Each stability metric was found to be significantly correlated with at least one indicator. However, two potential indicators (*ACBO* and *ARFC*) were not found to have significant correlation with any stability metric. Indicators *MIF*, *PF*, and *NACR* were found to have significant correlations with the largest number of stability metrics; each is significantly correlated with 10 out of the 17 stability metrics. *NSC* and each of the *NSxR* metrics have significant correlations with the largest number of indicators. *NSC* is significantly correlated with 9 out of the 16 indicators, and each of the *NSxR* metrics is significantly correlated with 8 out of the 16 indicators.

4.5.3. Multivariate analysis. Multivariate analysis was performed to construct different multivariate regression prediction models for each individual dependant variable (stability metric); each with a different subset of the independent variables (potential stability indicators). For each dependant variable, five multivariate regression models were built: Model 1 from C&K metrics only; Model 2 from MOOD metrics only; Model 3 from all product-related indicators (i.e. C&K metrics and MOOD metrics); Model 4 from process-related indicators; and Model 5 from all indicators (product-related and process-related). A stepwise selection process [9] that involves both forward selection and backward elimination was used to construct these models under a 95% confidence level.

Table 4 lists the independent variables in each one of the five constructed regression models for each dependant variable. In this table, 'N/A' implies that no significant regression model could be built to predict the corresponding dependant variable. Table 5 lists the coefficient of determination (R^2) for each kind of model per each dependant variable.

Comparing Models 1 (C&K metrics) with Models 2 (MOOD metrics) yields the following observations. Model 1 was constructed for 11 out of the 17 stability metrics, whereas Model 2 was constructed for 10 out of the 17 stability metrics. Model 1 is available for *NAC* and *NDC* metrics but not Model 2. Model 2 is however available for *NAAR* metric but not Model 1. Neither Model 1 nor Model 2 could be constructed to predict *NMC*, *NAGR*, *NADR*, *NASR*, *TBR* metrics.

Comparing Models 3 (product-related indicators) with Models 4 (process-related indicators) yields the following observations. Model 3 was constructed for 12 out of the 17 stability metrics, whereas Model 4 was constructed for 14 out of the 17 stability metrics. Model 3 is available for *NAC*, *NDC* and *NDDR* metrics but not Model 4. Model 4 is however available for *NMC*, *NAGR*, *NADR*, *NASR*, and *TBR* metrics but not Model 3. Both Model 3 and Model 4 or at least one of them could be constructed to predict each one of the stability metrics.

Model 5 (all indicators) is better than the other four types of models. Unlike the other four types of models, Model 5 could be constructed to predict each one of the stability metrics. In general, its coefficient of determination (R^2) is very high, where 12 out of the 17 models of type Model 5 have $R^2 > 90\%$.

Table 4. Models' independent variables

Dependent variable	Independent variables				
	Model 1	Model 2	Model 3	Model 4	Model 5
NSC	ANOC	PF	ANOC	AGE NPCR	AGE NPCR
NAC	ADIT	N/A	ADIT	N/A	ADIT NCCR
NDC	ALCOM	N/A	ALCOM	N/A	ALCOM
NMC	N/A	N/A	N/A	NPCR	NPCR
NSGR	AWMC ADIT ANOC ARFC	AHF MIF	AWMC ADIT ANOC ARFC	AGE NCCR NPCR	AWMC ADIT ANOC ARFC
NAGR	N/A	N/A	N/A	AGE NPCR	ADIT NPCR
NDGR	ALCOM	AHF AIF	AHF AIF	NACR	AHF NACR
NSAR	ADIT ANOC ACBO ARFC	AHF MIF PF	ADIT ANOC ACBO ARFC	AGE NACR	ADIT ANOC ACBO ARFC
NAAR	N/A	PF	PF	NACR	ADIT AIF NACR NPCR
NDAR	ALCOM	AHF AIF	AHF AIF	NACR	ALCOM NACR
NSDR	ANOC	AHF MIF	AHF MIF	AGE	AHF MIF
NADR	N/A	N/A	N/A	NACR	AWMC NACR
NDDR	ALCOM	AIF PF	ALCOM MHF MIF	N/A	AWMC ALCOM MHF AGE
NSSR	AWMC ADIT ANOC ARFC	AHF MIF	AWMC ADIT ANOC ARFC	AGE NCCR NPCR	AWMC ADIT ANOC ARFC
NASR	N/A	N/A	N/A	AGE NPCR	ADIT NPCR
NDSR	ALCOM	MIF CF PF	MIF CF PF	NACR	ALCOM NACR
TBR	N/A	N/A	N/A	NACR	NACR

Table 5. Models' coefficient of determination

	Model 1	Model 2	Model 3	Model 4	Model 5
NSC	69.33	64.10	69.33	96.04	96.04
NAC	39.88	N/A	39.88	N/A	81.37
NDC	35.27	N/A	35.27	N/A	35.27
NMC	N/A	N/A	N/A	92.72	92.72
NSGR	99.47	98.72	99.47	98.81	99.47
NAGR	N/A	N/A	N/A	85.81	90.26
NDGR	53.66	68.71	68.71	60.37	89.36
NSAR	99.24	98.46	99.24	95.95	99.24
NAAR	N/A	34.18	34.18	74.96	99.55
NDAR	60.89	70.25	70.25	59.01	91.10
NSDR	83.3	97.88	97.88	93.55	97.88
NADR	N/A	N/A	N/A	65.09	86.75
NDDR	64.69	74.98	90.30	N/A	97.67
NSSR	99.33	98.63	99.33	98.54	99.33
NASR	N/A	N/A	N/A	90.80	95.90
NDSR	61.11	85.47	85.47	58.77	91.08
TBR	N/A	N/A	N/A	50.37	50.37

5. Conclusions

This paper has empirically investigated potential product-related and process-related indicators of measures of structural stability of OODs through a case study of 13 successive releases of Apache Ant. The major findings of the case study reported in this paper can be summarized as follows. Each one of the stability metrics was found to be significantly correlated with at least one of the investigated indicators. In addition, it was possible to construct statistically significant regression models from subsets of the investigated indicators to early predict the value of each one of the stability metrics. Moreover, both product-related and process-related indicators should be considered in building stability prediction models.

As future work, additional case studies are needed to further support the findings of this paper. Another research direction is to construct and evaluate nonparametric prediction models such as neural network and Bayesian network that utilize the investigated indicators to predict the value of each one of the stability metrics.

6. References

[1] The Apache Ant Project, http://ant.apache.org/, (Accessed: 1 January 2005)

[2] ExamDiff Pro, http://www.prestosoft.com/ps.asp?page=edp_examdiffpro, (Accessed: 1 January 2005)

[3] JStyle, http://www.mmsindia.com/jstyle.html, (Accessed: 1 January 2005)

[4] B. Abreu and W. Melo, "Evaluating the Impact of Object-Oriented Design on Software Quality," *Proc. 3rd International Software Metrics Symposium*, pp. 90-99, 1996.

[5] J. Bansiya, "Evaluating Structural and Functional Stability," in *Building Application Frameworks: Object-Oriented Foundations of Framework Design*, M. Fayad, R. Johnson, and D. Schmidt, John Wiley & Sons, pp. 599-616, 1999.

[6] V. Basili and H. Rombach, "The TAME Project: Towards Improvement-Oriented Software Environment," *IEEE Transactions on Software Engineering*, vol. 14, no. 6, pp. 758-773, June 1988.

[7] S. Chidamber and C. Kemerer, "A Metrics Suite for Object Oriented Design," *IEEE Transactions on Software Engineering*, vol. 20, no. 6, pp. 476-493, June 1994.

[8] T. DeMarco, *Controlling Software Projects: Management, Measurement & Estimation*, Prentice-Hall, 1982.

[9] T. Dielman, *Applied Regression Analysis for Business and Economics*, 3rd Edition, Duxbury/Thomson Learning, 2001.

[10] D. Grosser, H. Sahraoui, and P. Valtchev, "An analogy-based approach for predicting design stability of Java classes," *Proc. 9th International Software Metrics Symposium*, pp. 252-262, 2003.

[11] D. Grosser, H. Sahraoui, and P. Valtchev, "Predicting Software Stability Using Case-Based Reasoning," *Proc. 17th IEEE International Conference on Automated Software Engineering*, pp. 295-298, 2002.

[12] R. Harrison, S. Counsell, and R. Nithi, "An Evaluation of the MOOD Set of Object-Oriented Software Metrics," *IEEE Transactions on Software Engineering*, vol. 24, no. 6, pp. 491-496, June 1998.

[13] M. Jazayeri, "On Architectural Stability and Evolution," *Lecture Notes in Computer Science, Springer-Verlag*, pp. 13-23, 2002.

[14] M. Mattsson and J. Bosch, "Characterizing Stability in Evolving Frameworks," *Proc. Technology of Object-Oriented Languages and Systems*, pp. 118-130, 1999.

[15] M. Mattsson and J. Bosch, "Stability assessment of evolving industrial object-oriented frameworks," *Journal of Software Maintenance: Research and Practice*, vol. 12, pp. 79–102, 2000.