Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2000 Proceedings

Americas Conference on Information Systems (AMCIS)

2000

Software Volatility: A Stystem-Level Measure

Evelyn Barry
Carnegie Mellon University, eb48@andrew.cmu.edu

Chris F. Kemerer
University of Pittsburgh

Sandra A. Slaughter
Carnegie Mellon University

Software Volatility: A System-Level Measure

Evelyn Barry, Carnegie Mellon University, Graduate School of Industrial Administration,
eb48@andrew.cmu.edu
Chris F. Kemerer, University of Pittsburgh, Katz School of Business
Sandra A. Slaughter, Carnegie Mellon University, Graduate School of Industrial Administration

Abstract

With change our only constant, information systems researchers appreciate the need to measure and understand change processes occurring in software systems (i.e., *software volatility*). In this study we define a system-level multi-dimensional measure of software volatility. This measure can be used both quantitatively and qualitatively to analyze system behavior. We describe the lifecycle volatility of three application systems. We also discuss use of software volatility as a qualitative measure to interpret system behavior for software portfolio management.

Keywords: *software volatility, software change, software lifecycle, software measurement, software maintenance*

Introduction

We have all heard the adage "The only thing constant is change". As researchers in information systems we recognize the need to measure and understand change processes occurring in software systems (i.e., *software volatility*). Software changes occurring over system lifecycle maintenance have traditionally been tracked and analyzed at the program level. To more fully understand lifecycle transformations occurring in software systems a more in-depth approach is required. In this study we define a system-level multi-dimensional measure of software volatility. This volatility measure can be used both quantitatively and qualitatively to analyze system behavior. We demonstrate its use by describing the lifetime volatility of three application systems. We also discuss use of software volatility as a qualitative measure for managers to categorize and interpret system behavior.

Defining Volatility

Software volatility is a measure of lifecycle changes occurring in software systems. Traditional system-level versioning fails to track the size or frequency of software changes. Measures of software modification size, or counts of modifications over time, are usually maintained at the program or module level. None of these measures allow direct comparison across systems of varying size and/or age. We seek a system-level volatility measure expressed as a non-negative number with scale invariance. To allow comparison across systems the measure also needs an upper bound (Allison, 1978). To find such a measure of software volatility we examine measures used in other fields. In their discussion of environmental variation, Wholey and Brittain (1989) describe three dimensions of environmental change: frequency, amplitude and predictability of variation. We apply these dimensions to software volatility.

Amplitude describes the magnitude of change. It can be expressed as the size of system modifications. We establish a system-level measure of amplitude as the change in application system size. A number of size metrics are available, including lines of code (LOC) and function points (Boehm, 1984; Albrecht and Gafney, 1983; Grady, 1987; Symons, 1988). Normalized size measures are needed to provide a bounded measure and allow comparison of amplitude, e.g. [change in LOC / total LOC]. Amplitude and normalized amplitude can be measured at regular time intervals throughout a system's lifetime.

Frequency describes time intervals between change events[1]. We begin by providing a definition of mean time since software modification, MTSM. This measure is similar to measures of mean time between failures found in manufacturing and software reliability engineering. We are examining intervals between software modifications regardless of the motivation for those modifications. Software reliability studies only measure intervals between software failures (Lyu, 1995). We define MTSM as the mean of the intervals between change events occurring in a system. Normalized MTSM, NMTSM, is the ratio of MTSM to the age of the application system. MTSM and NMTSM can be calculated at regular time periods throughout a system's lifecycle.

Predictability describes the probability that software modifications occur at the frequency described by NMTSM. Variance and standard deviation can be used in this way. However, they are not scale invariant. Predictability can be expressed as a measure of relative dispersion, or the dispersion of a variable about its mean value. The family of Gini indices for social-evaluation functions measures relative dispersion. The coefficient of variation allows comparison of dispersions for widely

[1] A change event is a recorded software modification or program creation.

varying groups of data, but it is not bounded. By definition, the Gini coefficient ranges from zero (no dispersion) to one (maximum dispersion) (Donaldson and Weymark, 1980). We use the Gini coefficient to represent the degree of irregularity in the overall pattern of environmental change. This measure is particularly useful for the analysis of software volatility, as it relates to a measure of the predictability of behavior among the programs in a software system. The Gini coefficient can be calculated for each time period in a system's lifecycle. We summarize the dimensional measures of volatility in Table 1.

Empirical Evaluation of Measures

Our research site is a large mid-Western retailer that has a portfolio of 23 legacy systems under maintenance. Over the course of the portfolio's 20-year history, the maintainers kept a detailed log of every modification made by recording date, purpose and type of change (Kemerer and Slaughter, 1999). Changes in application system size were calculated for each month in the portfolio lifecycle. We calculated Normalized LOC (NORMLOC), NMTSM and Gini coefficient for each month of each application's lifecycle.

Correlations of these dimensions of volatility are presented for three application systems in Table 2.What do these correlations tells us? *Financial system:* This is the oldest of the three applications with 246 months (20+ years) of productive life. There is a significant association among the measures of volatility. Larger modifications (NORMLOC ↑) are associated with shorter change intervals (NMTSM ↓) and a wider dispersion of volatility (Gini ↑).
Manifest system: This application is 10+ years old. Amplitude (NORMLOC) does not have a significant association with either frequency (NMTSM) or predictability (Gini coefficient). There is a significant association between frequency and predictability demonstrating that when change intervals decrease (NMTSM ↓), dispersion of volatility increases (Gini ↑).
Order Management system: This application is also 10+ years old. The only significant association in this case is that between amplitude and predictability. For Order Management, larger modifications (NORMLOC ↑) are associated with wider dispersion of volatility (Gini ↑).

These correlations provide a holistic view of the lifecycle software volatility. How could this information be used to describe differences in behavior as systems age? One approach would be to show the movement of a system through a space defined by the dimensions of volatility. We begin by creating a 2-dimensional space defined by frequency (NMTSM) and predictability (Gini coefficient). Movements were smoothed by classifying frequency as high/low for each month by comparing respective values with the lifecycle dimension means for each application. These categorical data were examined with non-parametric sequence analysis and gamma analysis techniques[2] (Pelz, 1985). The resulting gamma analyses are displayed as Table 3.
Financial System: Gamma analysis identifies 3 phases of system volatility behavior[3]. The initial phase shows a stable system with predictably long change intervals. The second phase shows high volatility with short change intervals and unpredictable behavior among the programs in the system. In the third phase, the behavior of the system becomes more predictable although the system continues to experience short change intervals.
Manifest system: Gamma analysis identifies 4 phases of system volatility behavior. As with the Financial system, the initial phase shows a system with predictably long change intervals. The second phase is identified as *pending*, indicating that system behavior is so inconsistent that sequence and gamma analysis is unable to clearly identify a prevalent behavior classification. The third phase indicates predictable short change intervals. The fourth phase identifies a system with unpredictable and frequent modifications.
Order Management system: Gamma analysis identifies 4 phases for this application. In contrast to the other systems, Order Management starts with predictable but short change intervals. The second phase continues to show predictable behavior with long change intervals. The third phase indicates long change intervals but with a high Gini coefficient designating a wide dispersion of change interval length among the system's programs. System behavior deteriorates in the last phase that is identified as pending.

The next step is to add amplitude as a third dimension creating 2 layers of these same categories with either high or low amplitude classifications. We can then study the progress of a system through the defined 3-dimensional space as it evolves.

Discussion

Clearly lifecycle patterns of behavior differ among these three applications. Use of non-parametric sequence and gamma analysis can be used for high-level interpretation of differences in system behaviors. Further investigation of application characteristics and other exogenous variables can contribute to our understanding of the evolutionary processes involved in these behaviors. By using quantitative system-level measures of software volatility with parametric analyses we can increase our ability to predict software system lifecycle behavior. With further analysis, we can use these measures to

[2] *Winphaser©* software is used for sequence analysis mapping procedures.
[3] A phase is identified by the software as 3 consecutive time periods (i.e. months) of matching behavior classification.

identify lifecycle stages in application systems and more precisely identify the best time to repair or replace a software system.

References

Allison, Paul D., "Measures of Inequality", *American Sociological Review*, Vol. 43, December 1978, pp. 865-880.

Donaldson, David and Weymark, John A., "A Single-Parameter Generalization of the Gini Indices of Inequality", *Journal of Economic Theory*, 1980, Vol. 22, pp. 67-86.

Kemerer, Chris F. and Slaughter, Sandra A., "An Empirical Approach to Studying Software Evolution", *IEEE Transactions on Software Engineering*, Vol.25, No. 4, 1999, pp.1-17.

Lyu, Michael R., *Handbook of Software Reliability Engineering*, IEEE Computer Society Press, Los Alamitos, CA, 1995.

Pelz, Donald C., "Innovation Complexity and the Sequence of Innovating Stages", *Knowledge, Creation, Diffusion, Utilization*, Vol. 6, No. 3, March 1985, pp. 261-291.

Wholey, Douglas R., and Brittain, Jack, "Characterizing Environmental Variation", *Academy of Management Journal*, Vol. 32, No. 4, 1989, pp. 867-882.

	LOC	Function Points	Normal-ized size	MTSM	NMTSM	Gini coefficient	Coefficient of variation
If individual elements all = 0, so does the measure	true	true	true	true	true	true	true
If any element > 0, then the measure > 0	true	true	true	true	true	true	true
Scale invariant, i.e. relative value doesn't change if unit of measurement changes	true	true	true	true	true	true	true
Bounds	(0, +∞)	(0, +∞)	(0, 1)	(0, +∞)	(0, 1)	(0, 1)	(0, +∞)

Table 1: Evaluating Measures of Volatility:

Application	N	(NORMLOC, NMTSM) (p value)	(NORMLOC, Gini) (p value)	(NMTSM, Gini) (p value)
Financial System	246	-0.2405 (0.0001)	0.2253 (0.0004)	-0.7164 (0.0000)
Manifest System	122	-0.1486 (0.1024)	-0.0370 (0.6858)	-0.4828 (0.0000)
Order Management System	125	-0.1712 (0.0563)	-0.1891 (0.0347)	-0.0601 (0.5057)

Table 2: Correlation of dimensions of Lifecycle Software Volatility

Application	1st phase	2nd phase	3rd phase	4th phase
Financial system	High NMTSM, low Gini coefficient	Low NMTSM, high Gini coefficient	Low NMTSM, low Gini coefficient	
Manifest System	High NMTSM, low Gini coefficient	Pending	Low NMTSM, low Gini coefficient	Low NMTSM, high Gini coefficient
Order Management system	Low NMTSM, low Gini coefficient	High NMTSM, low Gini coefficient	High NMTSM, high Gini coefficient	Pending

Table 3: Gamma Analysis[4]

[4] All precedence and sequence scores were significant in the gamma analysis for these 3 application systems.