Lecture 7

Empirical Studies of Software Evolution: Change Types Adaptive, Corrective, and Perfective Changes

Agenda

- Presentation on Kemerer and Slaughter's paper
- Discussion on Kemerer and Slaughter's paper

Change Types

- Adaptive changes: add new functionality to a system
- Corrective changes: fix faults in the software
- Perfective changes: improve developer's ability to maintain the software without altering functionality or fixing faults

Today's Presenter

- Arasi Aravindhan

Problem Definition

- Identify and understand the phases of software evolution

What are the differences between [Eick et al, TSE 2001] and [Kemerer and Slaughter 1999]

- Kemerer's paper focused more on justifying & choosing an approach
- Building models that help us understand software evolution
- Data sets: abundant, detail data, unique rich data produced by following rigorous process.
- quantitative -- statistics, math focused, qualitative

Kemerer and Slaughter [TSE, 2001]

- It is an ***empirical study paper***
  - No concrete hypotheses. In fact, the goal of this type of research is to **identify hypotheses,** and to discover a theory of the *process* of software evolution
  - Often seen in grounded theory, ethnography, etc.
- It is also a ***survey paper*** of research methods in studying software evolution.

Sideway Discussion: How to write a good survey paper

1.1.6 Basili et al.

A relatively more recent study by Basili et al. examined 25 software releases of 10 different systems at NASA Goddard, including over 100 software systems totaling about 4.5 million LOC [8]. A focus of the study was to characterize the types of maintenance activities and examine both the total effort and the effort distributions across these maintenance projects. Data collection lasted 18 months. The study looked at the three Swanson maintenance change types (corrective, adaptive, perfective) and a set of maintenance activities for each. They found that error correction efforts, typically small changes, required significant isolation activity, while enhancements required more time on inspection and certification. Effort for design and coding and unit testing (CUT) were similar for the two types. Statistical analysis of these differences was necessarily limited. Pie charts on effort percentages by task type by system and on Swanson typology are presented. Similar to earlier work in new development, this paper argues against small releases, presumably due to scale economies [6], [5], [7].

Author	Publication	Methodology	Data	Dependent Variables	Statistical Test
Belady & Lehman (1976)	IBM Systems Journal	Field study	21 user-oriented releases	Release sequence numbers, system age, system size, number of system modules, complexity	Multivariate regression, Auto-correlation
Yuen (1985)	IEEE Conference on Software Maintenance	Field study	5,000 "components" over 19 months period, 3,000 KLOC	Priority class, originator's reference, release affected, component affected, machine affected, category of error discovered, response time	Chi-square, Contingency coefficient measure, Time series, T-statistic, Auto and cross-correlations, Poisson distribution
Yuen (1987)	IEEE Conference on Software Maintenance	Secondary data analysis	Modules from OS/360, OMEGA, EXECUTIVE, BD, B, DOS, CCSS systems	Cumulative modules handled, handle rate, fraction of modules handled, size, release interval, net growth	Runs test, Turning points test, Phase length test
Yuen (1988)	IEEE Conference on Software Maintenance	Field study	"notices" - information issued to the commercial users of the system using same data set as Yuen (1985)	Releases and number of "notices" per week	Runs test, Turning points test, Phase length test, Time series analysis/Spectral analysis techniques, Linear filtering
Tamai & Torimitsu (1992)	IEEE Conference on Software Maintenance	Survey	95 systems from various organizations reporting on work done in prior 5 years. Mainframe software, 70% COBOL.	Age of software life span, software size before and after replacement, application areas, replacement factors	Sample statistic, Correlation
Cook and Roesch (1994)	Journal of Systems and Software	Field study	10 versions of real time German telephone switching software released over 18 months.	Number of functions, number of functions changed, number of major changes	Correlations, exploratory factor analysis with varimax rotation
Gefen & Schneberger (1996)	IEEE Conference on Software Maintenance	Field study	29 months of Software Problem Reports (SPRs), 250 KLOC	Modification type (total number of SPRs, number of corrective SPRs, number of adaptive SPRs), number of new applications, number of modifications caused by previous modifications	Linear regression, Wilcoxon Matched-Pairs Signed-Ranks Test, Kolmogorov-Smirnov Goodness of Fit Test
Basili, et al. (1996)	IEEE International Conference on Software Engineering (ICSE)	Field study	25 software releases of 10 different systems at NASA Goddard.	Effort and size for different types of maintenance activities/tasks.	Mann-Whitney U non-parametric test; OLS regression
Lehman, et al. (1997)	International Software Metrics Symposium	Field study	21 software releases of a financial package	Size of system in modules and number of modules changed	Least squares and inverse square regression model; mean absolute error

=> (1) **Longitudinal** study of (2) **rich** data from actual, business systems in real organization is required.

Study Approach

- Data Collection & Data Transformation
- Identify Measurement Variables
- Analysis
  - Time Series Analysis
  - Sequence Analysis
  - Phase Mapping

Study Approach: (1) Data Collection

- Some retailer system written in Cobol
- Stability of development & management teams
- 20 years of data collection
- 25000 change log events
  - module, author, function, date
  - english description of the change

Study Approach:
(2) Data Transformation

- Three independent coders manually coded change log data using content analytic approach
- To maximize inter-rater reliability,
  - A standard coding procedure was developed.
  - Several trial data coding processes were performed and Cohen's K measure improved 0.42 to 0.78.

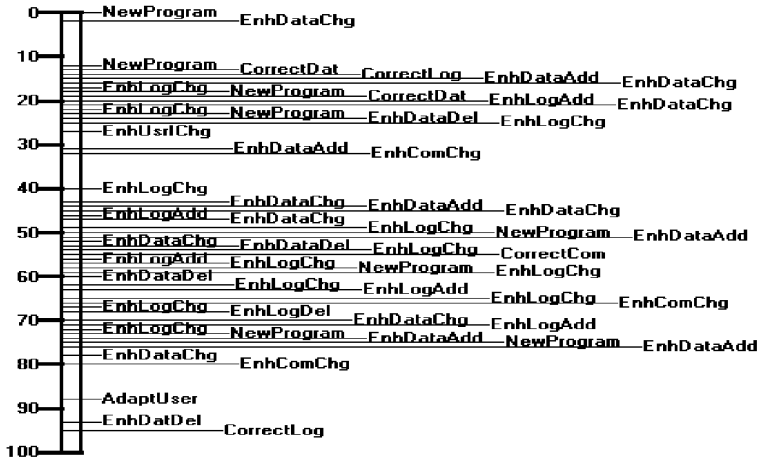
Study Approach:
(3)-1 Time Series Analysis

- Identification of a quantified, continuous variable:
  - # of changes per time period
    - Per year => not sufficient number of time periods
    - Per week or day => not sufficient number of changes per time period
- ARIMA (autoregressive integrated moving average)
- Data were not stationary
- Data series occurred in a largely random fashion

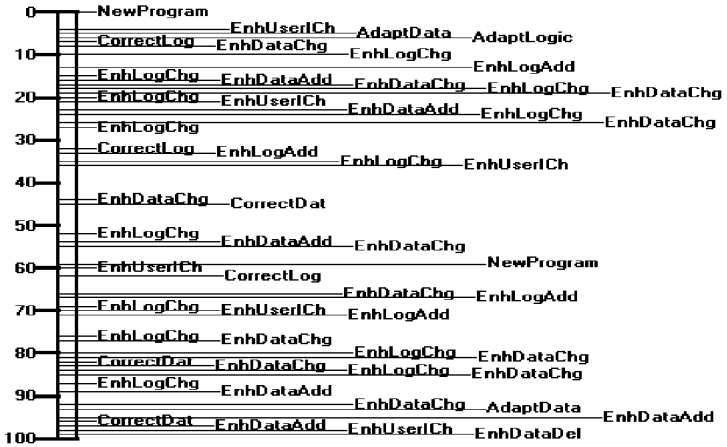
Study Approach: (3)-2 Sequence Analysis used in Social Science

- Identification sequences of acts or phases from a long series of categorical data
- Phase mapping
  - group by four consecutive events of the same type
- gamma sequence analysis -- Goodman-Kruskal score that assesses the proportion of phase order

Results from Sequence Analysis



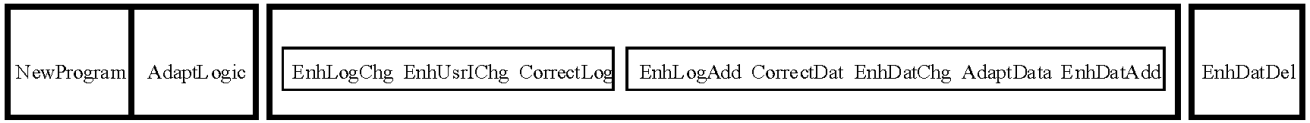
Financial Sales System



Manifest Shipping System

Results from Sequence Analysis -2

Gamma map for financial sales reporting system.



). Gamma map for manifest shipping system.

Threats to Validity

- External Validity: Does this work generalize to other situations?
- Construct Validity: Was the manual labeling / categorization process reliable and reproducible? Did programmers follow a rigorous data collection procedure?
- Conclusion Validity: Would the authors get the same results by grouping events with n consecutive events instead of 4 and by using a different parameter for phase analysis?

Any surprising results?

- More than 60% were enhancement and perfective changes.
-

My general thoughts on Kemerer and Slaughter [TSE, 1999]

- A highly disciplined qualitative study approach
- Attempt to understand software evolution processes from bottom-up using qualitative methods
- Nice attempt to incorporate change types
- In-depth, longitudinal study of two systems
- Results are hard to understand --- It is difficult to find any meaningful conclusions from the phase maps and gamma tables