Presentation of
“Identifying reasons for software change using historic databases.”

Lucas Panjer

October 19, 2006

Hypothesis

- Studies have revealed three major causes of software change
  - Adding features (adaptive)
  - Correcting faults (corrective)
  - Restructuring (perfective)
- Textual description of a change is tightly correlated to type of change described
- Difficulty, size and interval of changes vary across different types of changes

Experiment

- Quantitative
  - Developed algorithm to classify textual change messages into three categories
  - Verification: manual survey by developers, to classify recent changes
  - Test on multiple products

System A

- Characteristics
  - 2M LOC
  - 3000 files
  - 100 modules
- Over last 10 years
  - 33171 MR
  - Avg 4 deltas / MR

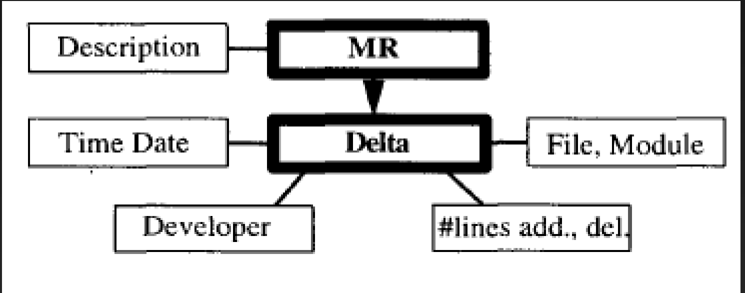

Figure 1. Changes to the code (bold boxes) and associated attributes. Each modification request contains a number of deltas.

Classification of Maintenance Activities

- Normalization (fixes, fixing, fix => fix)
- Word Frequency
  - Corrective: correct, fix, problem,…
  - Prescriptive: add, new, modify, update,…
  - Perfective: cleanup, unneeded, remove,…
- Keyword clustering

Classification Process

- Sequence of rules
  1. Inspection class matched first
  2. Presence of a keyword classifies a MR
  3. Multiple keywords are resolved to the most common type

Results

- 45% Adaptive changes
- 34-46% Corrective changes 18-27% of LOC
- Inspection changes had largest LOC
- Perfective changes delete most lines / delta

Validation

- Develop surveys
- Classify recent MRs
- 5 developers, 30 MRs each
- Developer and classifier agree 61% of the time

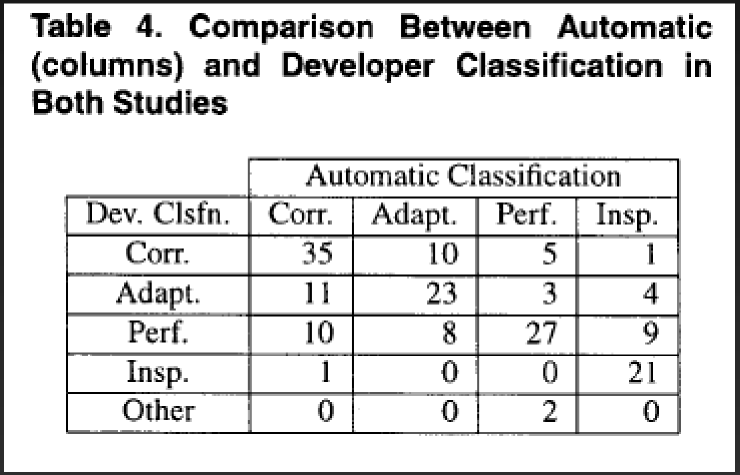

Table 4. Comparison Between Automatic (columns) and Developer Classification in Both Studies

	Automatic Classification			
Dev. Clsfn.	Corr.	Adapt.	Perf.	Insp.
Corr.	35	10	5	1
Adapt.	11	23	3	4
Perf.	10	8	27	9
Insp.	1	0	0	21
Other	0	0	2	0

Change purpose related to size and interval

- Most time consuming 35% of Adaptive changes took much longer than the 35% of Inspection changes
- New code (Adaptive) and Inspections add the most LOC
- Perfective removes the most LOC

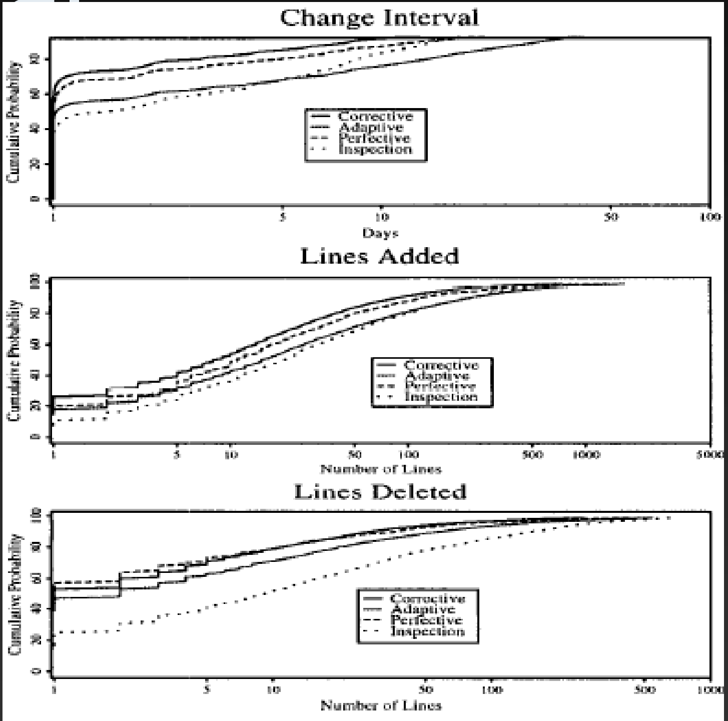

Figure 2. The three plots compare empirical distribution functions of change interval, added lines, and deleted lines for corrective (solid line), adaptive (dotted line), perfective (dashed line), and inspection (long-dashed line) types of changes. Adaptive changes add the most code and take the most time to complete. Inspection changes delete the most code, and corrective changes take the least time to complete.

System A/B Comparison

- System B
  - Very similar to System A
  - Different developers, sub-organization
  - Same VCS tool model
- Differences are minimal
- Not explained by analysis

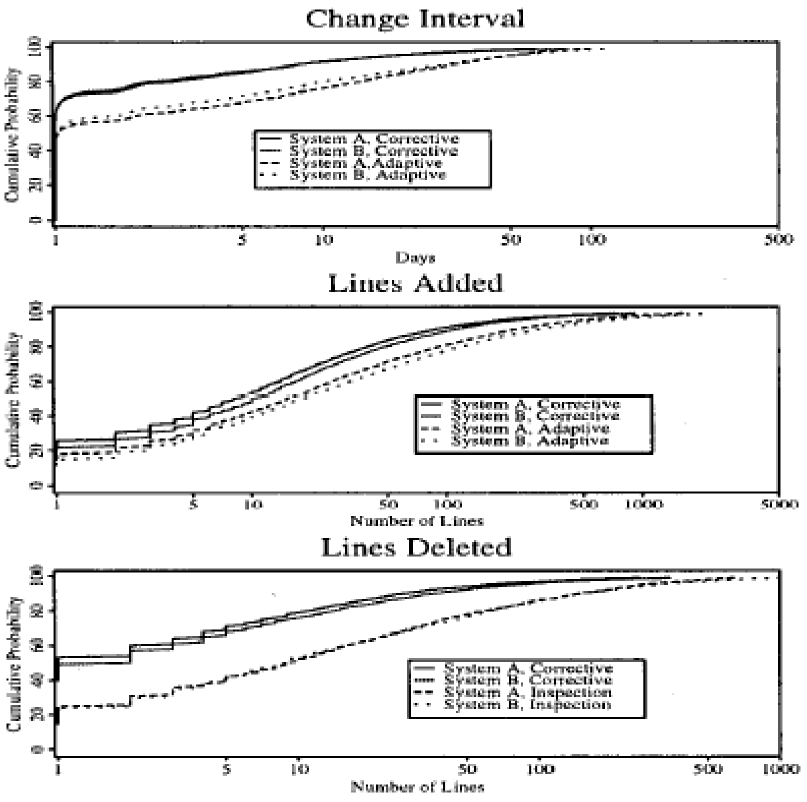

Figure 4. Comparison of two products in terms of empirical distribution functions of change interval and numbers of added or deleted lines for corrective, adaptive, perfective, and inspection changes.

Change difficulty

- Corrective changes rated hard most often
- Size, corrective maintenance, developer are important variables

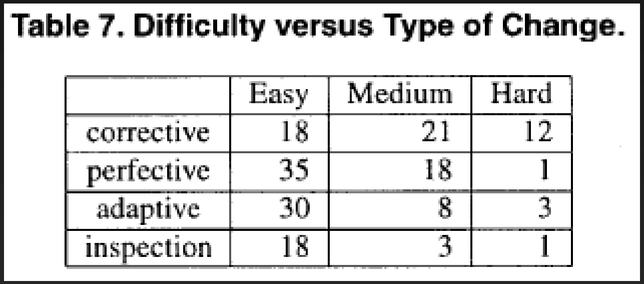

Table 7. Difficulty versus Type of Change.

	Easy	Medium	Hard
corrective	18	21	12
perfective	35	18	1
adaptive	30	8	3
inspection	18	3	1

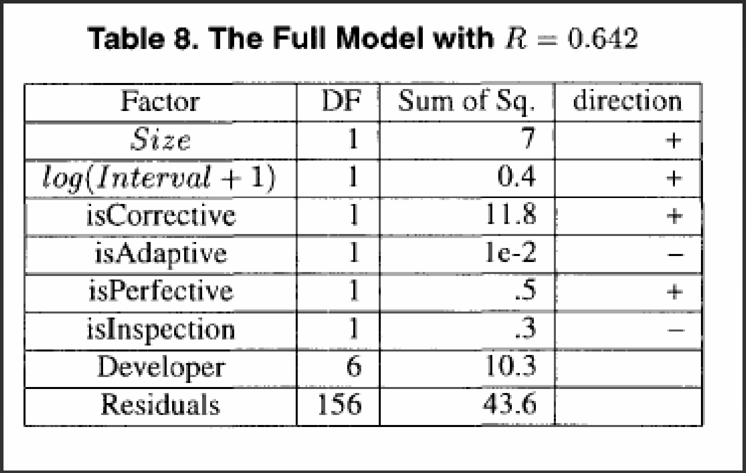

Table 8. The Full Model with $R = 0.642$

Factor	DF	Sum of Sq.	direction
$Size$	1	7	+
$log(Interval + 1)$	1	0.4	+
isCorrective	1	11.8	+
isAdaptive	1	1e-2	–
isPerfective	1	.5	+
isInspection	1	.3	–
Developer	6	10.3	
Residuals	156	43.6	

Contributions

- Predictor shows little variance between products
  - Indicates that size and interval of change might be used to identify the reason for a change
  - Indicates a possibility of broader generalization (external validity)
- Recommends features for future version control systems to aid analysis and recovery of software evolution
- Basic knowledge of software change types and attributes

Positive

- Able to derive a lot of knowledge out of a simple VCS model
- Developed a general technique and infrastructure to apply this technique to other projects
- Good prediction accuracy
  - 61% for type of change based on text

Negative

- Not able to classify all changes
- Limited to large telecom projects in same organization
- Much more information usually available which was not considered