Improvement of The Fault-Prone Class Prediction Precision by The Process Metrics Use

Nobuko Koketsu, N.Honda, S.Kawamura, J.Nomura
NEC Corporation
Makoto Nonaka
Toyo Univ.
## Business Domains and Our Chief products

### IT Services
- Cloud-Oriented Service Platform Solutions
  - MegaOak
  - PanelDirector
-Carrier Network
  - Long Term Evolution Network Systems
  - WiMAX Network Systems
- Social Infrastructure
  - Unity Cable Systems
  - Compact Microwave Communications Systems
  - Digital Terrestrial TV Transmitters
  - Asteroid Explorer "HAYABUSA" (provided by Japan Aerospace Exploration Agency)

### Platform
- Super Computer
- WebSAM
- Integrated Operation/Management Middleware
- UNIVERGE
  - Unified Communication

### Personal Solutions
- Personal Computers
- Mobile Terminals

### Others
- Electron Devices
- Lithium-ion Batteries
- Liquid Crystal Displays
Characteristics of organization

- Continuous software quality improvement activity for more than 20 years

- Quality management with highly matured software life cycle process with technique known as “Quality accounting”

- Attained CMMI level 5 in 2004.

No. of Defects 1/20
Co-operative Development

Products are made by local development companies and overseas operations
The Process Phase of Software Development

The process phase: defined as the product development and quality improvement process.

Each phase defines deliverables, implementation tasks and measurement items

---

**Standardized development process in organized manner**

<table>
<thead>
<tr>
<th>Development department</th>
<th>Upstream process</th>
<th>Testing process</th>
<th>Release/Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product development</td>
<td>BD: Basic design</td>
<td>ST: System testing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FD: Functional design</td>
<td>FT: Functional testing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DD: Detailed design</td>
<td>UT: Unit testing</td>
<td></td>
</tr>
<tr>
<td>Quality improvement</td>
<td>CD: Coding</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quality improvement activities</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Quality Assurance Process

- Performed independently by the quality assurance department
- The data is collected at every phase of the development process
- Analyzed quality objectively and from various angles to identify any issues.

Typical Process Metrics

<table>
<thead>
<tr>
<th>Phased to be measured</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entire development process</td>
<td>Starting date delay (days), completion date delay (days), number of defect/KL, and rate of defect detection</td>
</tr>
<tr>
<td>Only upstream process</td>
<td>Rate of work progress, effort/KL, review effort/KL, and defect count/review effort</td>
</tr>
<tr>
<td>Only testing process</td>
<td>Execution ratio of test items, number of test items/KL, and testing effort /KL</td>
</tr>
</tbody>
</table>

Established quality assurance activity based on collected metrics
Issues in Current Quality Assurance

- Product is composed of sub systems

- Quantitative quality management is based on the collected data reported from developer

- Control with higher precision is left to individual quality analysis done by each developer.

  - If developer’s analysis skill is low, quality problems occur until the last phase

  - Analysis skill can not be assessed objectively
Solution

Need to identify quality status in higher precision

- No dependency on developer’s analysis
- In the quantitative value
- More precision than sub-system level
What is the Fault-Prone Class Prediction?

What is Fault-Prone (FP) Module Prediction?
- Method to identify any module that is more likely to contain a fault, from among all the modules constituting the software

Unit of fault-prone module*
- A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading.
- A logically separable part of a program

*(reference) Definition of “module” in IEEE Std 610
- File, Class, Function, Class Method etc…

Could be applied to solve our issues
The Situation of FP Module Prediction

The FP module prediction studies are becoming widespread.
- Initial research started in the 1980s
- Use CK metrics as an explanatory variable for object-oriented software in the 1990s.
- Study for OSS or excluding the dominant multiple logistic regression model used in the past.

Explanatory variables of FP module prediction
- Metrics collected from source code (most popular)
  - Scale … LOC
  - Complexity level … Cyclomatic complexity
  - Design … CK object-oriented metrics, Fan-in/out

- Information for design documents (less study)
  - Describe a design element graphically and extract it automatically

- Factors related to the process (less study)
Issues in The Application of FP Prediction Study

Our organization and data does not meet the precondition of the analysis of existing study of FP prediction

- Fault number in module to be predicted with multiple regression model of product metrics and process metrics (Shen 1985)
  → Development language, used metrics and fault density in test phase and post-release

- Fault-Prone model prediction with automatic extraction of design information from the specification (Ohlsson 1996)
  → It is difficult to change existing design methodology only for FP prediction

- Predict Fault-Prone Class with CK metrics (Dr. Basili 1996)
  → Target is software development project by students

  → It is difficult to apply open source data to the development of commercial software under established quality assurance process

- Fault number to be predicted with process data (qualitative data) as variant of Bayesian network (Fenton 2007)
  → Different nature from quantitative data measured in the organization
Application to Actual Development

- We built the environment that can automatically measure scale and complexity of source code under development. It is possible to collect metrics used for FP module predication during development.

- The modification of source code that happened after functional testing phase are controlled.
  - It is possible to apply model based on modification after functional test and collected metrics.

- Effective in improving precision of quality assurance than doing it per subsystem.
  - c.f. 105KLoc source code subsystem 7: class 215 -> nearly 30 times

Application of FP module prediction to improve the precision of existing quality assurance method.
Approach to Apply FP Class Prediction

Using metrics which collected by coding and unit testing phase, we predict the class with the possibility of the modification after functional testing

- Use as candidate of explanatory variables
  - CK metrics
  - Process metrics
  - Automatically measurable metrics

Modification occurred

Process metrics
Use for quality management

• CK metrics
• Automatically measurable metrics
Collected by source code
Candidate for Explanatory Variable 1

### Process metrics
- Collected until unit testing

<table>
<thead>
<tr>
<th>Phased to be measured</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entire development process</td>
<td>number of defect/KL, and rate of defect detection</td>
</tr>
<tr>
<td>Only upstream process</td>
<td>effort/KL, review effort/KL, and defect count/review effort</td>
</tr>
<tr>
<td>Only testing process</td>
<td>execution ratio of test items, number of test items/KL, and testing effort /KL</td>
</tr>
</tbody>
</table>
Candidate for Explanatory Variable 2

Automatically measurable metrics collected from source code

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scale</strong></td>
<td></td>
</tr>
<tr>
<td>Number of effective lines</td>
<td>Value derived by subtracting the comment and blank lines from the total number of lines (total summation per class)</td>
</tr>
<tr>
<td>Number of methods</td>
<td>Number of methods contained in class</td>
</tr>
<tr>
<td><strong>Complexity level</strong></td>
<td></td>
</tr>
<tr>
<td>Cyclomatic complexity</td>
<td>Value representing route complexity by branching command (total summation per class)</td>
</tr>
<tr>
<td>Number of branch conditions</td>
<td>Number of conditional equations for branching command (total summation per class)</td>
</tr>
<tr>
<td>nesting levels</td>
<td>Class average of maximum number of nesting levels for each method</td>
</tr>
</tbody>
</table>
Candidate for Explanatory Variable 3

CK metrics: *Reported as effective* for FP module prediction

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMC</td>
<td>Weighted Methods per Class</td>
</tr>
<tr>
<td>DIT</td>
<td>Number of hierarchies up to root class in inheritance tree (<strong>Depth of Inheritance Tree</strong>)</td>
</tr>
<tr>
<td>NOC</td>
<td>Number of direct subclasses (<strong>Number Of Children</strong>)</td>
</tr>
<tr>
<td>CBO</td>
<td>A count of the number of non-inheritance related couples with other classes (<strong>Coupling Between Objects</strong>)</td>
</tr>
<tr>
<td>RFC</td>
<td>Total of methods in which an object is executed in response to received messages (<strong>Response For a Class</strong>)</td>
</tr>
<tr>
<td>LCOM</td>
<td>Number of methods in which common attributes are manipulated, which represents a lack of cohesion (<strong>Lack of Cohesion in Methods</strong>)</td>
</tr>
</tbody>
</table>
Outline of Analysis

- Collected from the initial development of a new product and from the first version-up development.
- When the first version-up developed, method for separating the phases was divided into two ways according to its subsystems.
- We use version-up data is divided into two.

### Characteristics of software used for analysis

<table>
<thead>
<tr>
<th>Reference symbol</th>
<th>Development content</th>
<th>Development size</th>
<th>Analysis size</th>
<th>Number of classes</th>
<th>Number of subsystems</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>New development</td>
<td>105KL</td>
<td>57KL</td>
<td>215</td>
<td>7</td>
</tr>
<tr>
<td>B</td>
<td>Version-up development1</td>
<td>177KL</td>
<td>165KL</td>
<td>611</td>
<td>9</td>
</tr>
<tr>
<td>C</td>
<td>Version-up development2</td>
<td>124KL</td>
<td>91KL</td>
<td>450</td>
<td>6</td>
</tr>
</tbody>
</table>
FP Class Predication Evaluation Patterns

1 (with fault) and 0 (without fault) were assigned to classes

9 evaluation patterns, depending on which data is used for the creation and evaluation of expression

- P1, P2, P3: creation and evaluation by using the same data
- P4, P5: really used pattern
- P6 to P9: no use

<table>
<thead>
<tr>
<th>Reference</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
<th>P6</th>
<th>P7</th>
<th>P8</th>
<th>P9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data for model creation</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>A</td>
<td>B</td>
<td>B</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Data for model evaluation</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>C</td>
<td>A</td>
<td>B</td>
</tr>
</tbody>
</table>
## Evaluation Indices and threshold for FP Predictions

### Evaluation Indices

<table>
<thead>
<tr>
<th>Evaluation indices</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall ratio</td>
<td>Ratio of modules correctly determined to be FP among those modules that were actually faulty</td>
</tr>
<tr>
<td>Precision ratio</td>
<td>Ratio of modules that were actually faulty among those modules determined to be FP</td>
</tr>
<tr>
<td>F value</td>
<td>Harmonic mean of recall and precision ratios. A larger harmonic mean represents a higher-precision determination</td>
</tr>
<tr>
<td>Early fault class test ratio</td>
<td>Original index. Average of the rate determined not to be FP among all classes, and a recall ratio</td>
</tr>
</tbody>
</table>

\[
F \text{ value } = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}
\]

Recall, precision and F value was defined by Kaur, A. and Malhotra, R.

"Early fault class test ratio" 0.55 and F value 0.4 are set as a threshold of model equation based on the experience and knowledge.
Evaluation of Logistic Regression Model

- **P1 to P3**: Results are high
- **P4, P5**: Both results are lower than threshold
Application of Bayesian Network Model

Linear Regression Model assume the absence of a mutually dependent relationship between explanatory variables

Try to predict with a Bayesian Network Model

What is the Bayesian Network Model?

- One type of network model that stochastically describes cause and effect relationships
- A probabilistic inference model that expresses inferences for relationships based on a directed graph
- Individual variable relationships based on a conditional probability
Evaluation method of the network model

To evaluate the precision of the prediction of the Bayesian network models

- We used a bootstrap method
- 80% of all the samples were randomly selected from data to create a expression.
- Then, 4000 iterations were made to evaluate the created expression
- The average obtained from these iterations was treated as the final predictive value.

### Characteristics of Bayesian network models to be evaluated

<table>
<thead>
<tr>
<th>Reference symbol</th>
<th>Model type name</th>
<th>Mutually dependent relationship between objective and explanatory variables</th>
<th>Mutually dependent relationship between explanatory variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>BN1</td>
<td>NaiveBayes</td>
<td>Required</td>
<td>Absence</td>
</tr>
<tr>
<td>BN2</td>
<td>TAN</td>
<td>Required</td>
<td>Presence (Max 1)</td>
</tr>
<tr>
<td>BN3</td>
<td>Bayes Net</td>
<td>Optional</td>
<td>Presence (Max 3)</td>
</tr>
</tbody>
</table>

TAN: Tree Augmented NaiveBayes
The Results of Naive Bays Evaluation

- P1 to P3: Results are high
- P4’s results are higher than threshold but P5’s results are lower than threshold

![Evaluation pattern graph]

- ■ - Precision ratio
- □ - F value
- ◇ - Early fault class test ratio

Evaluation index value

Evaluation pattern
The Results of Evaluation (Bayes Net, TAN)

- P4, P5: Both model’s results are higher than threshold
- In particular, all TAN model’s evaluation results are higher than threshold

Certain level of robustness could be ensured with TAN model
Contribution of Process Metrics

Without process metrics, the number of nonconformance expression in is larger, and the early fault class test ratios and F values are lower.

Process metrics contribute to the increase in the TAN model prediction precision.

Results of evaluation of TAN

Results of evaluation of TAN (no process metrics)
Creation of the expression to Actual Development

FP Class prediction upon the completion of coding phase.

<table>
<thead>
<tr>
<th>Accomplishment (subsequent to the functionality testing)</th>
<th>Results of FP class prediction</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>the absence of any modification</td>
<td></td>
</tr>
<tr>
<td>Absence of any modification in the steps</td>
<td>303</td>
<td>149</td>
</tr>
<tr>
<td>Presence of any modification in the steps</td>
<td>59</td>
<td>100</td>
</tr>
<tr>
<td>Total</td>
<td>362</td>
<td>249</td>
</tr>
</tbody>
</table>

To detect faults in 100 classes

- Without FP class prediction: **63%** (100/159) of test items to be executed
- With FP class prediction: **41%** (249/611) of test items to be executed

Possible to improve the quality at early stage of the testing phase
Future study

To increase the precision of the prediction with the accumulation of data

- Analyze the cause of generated modification well as the classification of those modification
  (A modification in specification level or a fault in coding is treated equally as 1 modification)

To improve the precision of existing quality assurance method

- Impact of Complexity to maintainability
- Identify “Quality” that is measurable with product metrics
Conclusion

- Predict modification classes that occur after functional testing using the FP class prediction.

- By applying the network model for prediction, we can construct an expression that ensures a certain robustness for actual products.

- Improve the robustness of the expression by adding process metrics to collect metrics from the source code.

- Fixed explanatory variable could not be identified, but can identify quality status in higher precision than subsystems.

- Proposed a method for improving the quality at early stage of the test phase by using the FP class prediction.
NEC Group Vision 2017

To be a leading global company leveraging the power of innovation to realize an information society friendly to humans and the earth.