# Using Software Birthmarks to Identify Similar Classes and Major Functionalities

Takeshi Kakimoto Akito Monden Yasutaka Kamei
Haruaki Tamada Masateru Tsunoda Ken-ichi Matsumoto
Nara Institute of Science and Technology
8916-5 Takayama Ikoma Nara Japan 630-0192
{takesi-k, akito-m, yasuta-k, harua-t, masate-t, matumoto}@is.naist.jp

## ABSTRACT
Software birthmarks are unique and native characteristics of every software component. Two components having similar birthmarks indicate that they are similar in functionality, structure and implementation. Questions addressed in this paper include: Which are similar class files? Can they be gathered into one class file? What are major functionalities among class files? To answer to these questions, this paper analyzed the similarity of birthmarks for all pairs of classes in ArgoUML, and visualized them using Multi-Dimensional Scaling (MDS). As a result, three pairs of very similar class files, which seem to be made by copy-and-paste programming, were identified. Also, four major functionalities were identified in the MDS space.

## Categories and Subject Descriptors
D.2.8 [**Software Engineering**]: Metrics – *Product Metrics*; K.6.3 [**Management of Computing and Information Systems**]: Software Management – *Software maintenance;*

**General Terms:** Measurement, Experimentation

## Keywords
software birthmark, multi-dimensional scaling

## TARGET OSS PROJECT
ArgoUML (written in Java)

## MINING AREA
- Change impact, propagation coupling analysis
- Architecture and design quality analysis

## MINING QUESTIONS
The following two questions are addressed in this paper.

(1) Which are similar class files?

This question needs to be answered when one wants to refactor a Java program. Similar class files are often refactored into one class file to improve software maintainability. Also, when one modifies a class file, he/she often needs to find similar class files that need to be modified as well.

(2) What are major functionalities among class files?

This question needs to be answered when one joins a project and tries to understand the mapping between class files and their functionalities.

## INPUT DATA
From 1,432 class files of ArgoUML release 0.20, 61 classes having more than 30 lines of source code were chosen as an input dataset.

## 1. APPROACH AND TOOLS USED

### 1.1 Birthmark
Java birthmarks[1] are unique and native characteristics of every Java class files. Originally, birthmarks are used to detect the stolen (i.e. illegally copied) Java class files across two different projects. In this paper we use birthmarks to find similar class files in a project to help maintenance activities.

We used a Java birthmark tool called *jbirth*[1] to extract four types of birthmarks from each class file: (1) constant values in field variables (*CVFV birthmark*), (2) sequence of method calls (*SMC birthmark*), (3) an inheritance structure (*IS birthmark*), and (4) used classes (*UC birthmark*). Two class files having similar birthmarks indicate that they are similar in functionality, structure and implementation.

To compute the similarity between two class files $p$ and $q$ in terms of their birthmarks, we used the following definition [1].

**Definition (Similarity)** Let $f(p) = (p_1, \ldots, p_n)$ and $f(q) = (q_1, \ldots, q_n)$ be birthmarks with length $n$, extracted from class files $p$ and $q$. Let $s$ be the number of pairs $(p_i, q_i)$'s such that $p_i = q_i$ $(1 \le i \le n)$. Then, similarity between $f(p)$ and $f(q)$ is defined by: $s/n \cdot 100$.

### 1.2 Multi Dimensional Scaling (MDS)
After computing the similarity of birthmarks for all pairs of 61 class files, we used MDS to visualize their relationships. Major functionalities can be identified as clusters in the MDS space. We used SPSS as a MDS tool.

**Table 1. Pair of classes having high similarity birthmarks.**

| Class pairs | similarity |
|---|---|
| uml.ui.behavior.collaborations.PropPanelCollaboration | 96.09 |
| uml.ui.behavior.use_cases.PropPanelUseCase | |
| uml.ui.behavior.collaborations.PropPanelMessage | 93.91 |
| uml.ui.behavior.state_machines.PropPanelTransition | |
| uml.ui.foundation.core.PropPanelClass | 92.78 |
| uml.ui.foundation.core.PropPanelAssociationClass | |

[1] http://se.naist.jp/jbirth/

Copyright is held by the author/owner(s).
*MSR'06*, May 22-23, 2006, Shanghai, China.
ACM 1-59593-085-X/06/0005.

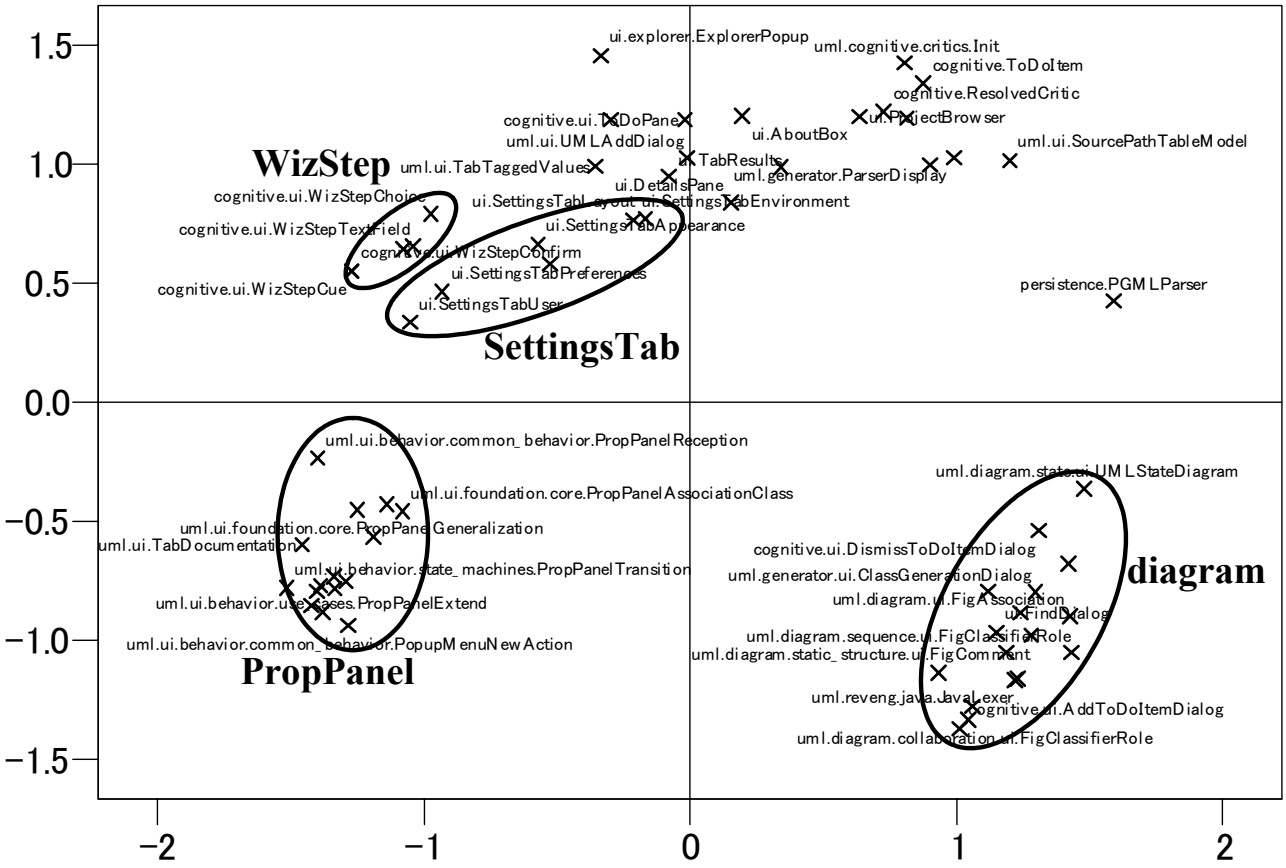


**Figure 1. Relationship among class files in MDS space based on similarity of birthmark.**

## 2. RESULTS AND INTERPRETATIONS

### 2.1 Finding Similar Class Files

Table 1 shows pairs of classes that had high-similarity birthmarks (similarity > 0.9). As we investigated their source code, each pair had very similar functionality, structure and implementation. It can be considered that these pairs were made by copy-and-paste programming, and can be refactored so as to reduce the duplicated code.

### 2.2 Finding Major Functionalities

Figure 1 shows relationship among classes in the MDS space. Classes having similar birthmarks are located in near space, and classes having dissimilar birthmarks are located far apart. From Figure 1, we could identify the following four major functionalities.

- Most of classes at the lower right part ("diagram" circle) were related to diagram (e.g. FigAssociation class, FigUseCase class.)
- Classes at the lower left part ("PropPanel" circle) were related to PropPanel (e.g. PropPanelAttribute class, PropPanelOperation class.)
- Classes related to WizStep were in "WizStep" circle (e.g. WizStepConfirm class.)
- Classes related to SettingsTab were in "SettingsTab" circle (e.g. SettingsTabEnvironment class.)

All these functionalities were identified by finding clusters and similar file names in the MDS space. We believe that using birthmarks together with MDS is useful to understand the relations between class files and to roughly recognize their functionalities.

## 3. CONCLUSIONS

This paper analyzed the similarity of birthmarks for all pairs of classes in ArgoUML, and visualized them using MDS. As a result, three pairs of very similar class files were identified. Also, four major functionalities (diagram, PropPanel, WizStep and SettingsTab) were identified in the MDS space.

## 4. ACKNOWLEDGMENTS

This work is supported by the EASE (Empirical Approach to Software Engineering) project of the Comprehensive Development of e-Society Foundation Software program of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

## 5. REFERENCES

[1] Tamada, H., Nakamura, M., Monden, A., Matsumoto, K. Java birthmark –Detecting the software theft. *IEICE Transactions on Information and Systems, E88-D*, 9 (Sept. 2005), 2148-2158