Object-Centric Instrumentation with Pharo

Steven Costiou

Square Bracket tutorials
February 19, 2019

Layout and typography based on the `sbabook` LaTeX class by Damien Pollet.

Contents

Illustrations

CHAPTER 1

Anonymous subclasses

Anonymous classes are nameless classes that are inserted between an object and its original class [FJ89, HJJ93]. The object is migrated to that new class, which takes the original object's class as its superclass. Methods from the original class can be redefined and reimplemented in the anonymous class, having the effect to change the behavior of that single object. Original behavior that is not redefined in the anonymous subclass is preserved. It is one of the fastest implementation for object-centric instrumentation [Duc99].

1.1 Example

Anonymous subclasses are derived from the original class of the object (line 3). Methods must be manually (re)written with instrumentation and compiled in the new class (line 4-8). Then the object has to be migrated to its new class (line 10). To rollback the instrumentation, the object must be manually migrated back to its original class (line 12). The migration is not *safe* if more than one process is using the instrumented object.

```
1  |person anonClass|
2    person := Person new.
3    anonClass := anObject class newAnonymousSubclass.
4    anonClass
5      compile:
6        'name: aName
7          self tag: aName.
8          name := aName'.
9    "migrates the object to its new class"
10   anonClass adoptInstance: person.
11   "migrates back the object to its original class"
12   anonClass superclass adoptInstance: person.
```

1.2 Evaluation

Manipulated entity: Classes. Behavioral variations are expressed in standard methods, compiled in anonymous classes.

Reusability: Partial. As an anonymous subclass is derived from the original class of an object, only instances of that same class can be migrated to the anonymous subclass. To apply the same instrumentation to an instance of another class, a new anonymous subclass must be created and the instrumented behavior must be recompiled in that subclass.

Flexibility: None. Instrumented methods must always be copied down to anonymous subclasses, and instrumentation must be inserted in the duplicated code. Without any tool built on top, that instrumentation is fully manual. Anonymous subclasses cannot be composed.

Granularity: Method. Instrumentation is implemented by recompiling modified copies of methods in anonymous subclasses. Sub-method level is achieved through manual rewriting of the method.

Integration: Partial. The object is migrated to an anonymous subclass, which does not break system tools. However, it is explicit that the object is now instance of an anonymous subclass. It may also break libraries and tools that use classes and class names as a discriminator.

`Self` problem. Solved by design: `self` always references the original object.

`Super` problem. Not solved. There are no means to express how to resolve the lookup when a message is sent to `super` from a method copied down in an anonymous subclass.

1.3 Other documentation

The Pharo Mooc provides materials on object-centric instrumentation based on object class migration and its flavours:

- http://rmod-pharo-mooc.lille.inria.fr/MOOC/Slides/Week7/C019-W7S04-OtherReflective.pdf
- http://rmod-pharo-mooc.lille.inria.fr/MOOC/WebPortal/co/content_78.html

Bibliography

[Duc99] Stéphane Ducasse. Evaluating message passing control techniques in smalltalk. *Journal of Object Oriented Programming*, 12:39–50, 1999.

[FJ89] Brian Foote and Ralph E Johnson. Reflective facilities in smalltalk-80. In *ACM Sigplan Notices*, volume 24, pages 327–335. ACM, 1989.

[HJJ93] Bob Hinkle, Vicki Jones, and Ralph E Johnson. Debugging objects. In *The Smalltalk Report*. Citeseer, 1993.