Give a man a dynamically typed language, will he write dynamically typed code?

A type system is a tractable syntactic method for proving the absence of certain program behaviors by classifying phrases according to the kinds of values they compute. Statically typed languages, such as C or Java, force the developer to specify within the source code the type of all variables, parameters and class members, either explicitly or implicitly. On the other hand, dynamically typed languages remain agnostic to variable types at compile time, and only take them in to account at runtime. This gives more freedom to the developer, as one variable can take on radically different types during program execution, if that is what the developer needs. Lets look at the following example in smalltalk, which does not have a syntactically correct equivalent in statically typed languages.

```
| a |
a := 42.
a:= 'String'.
a := Dictionary new.
```

We see a variable named a, and an assignment of three radically different objects (an Integer, a String and a Dictionary). This is perfectly correct smalltalk syntax, and the freedom to do this differentiates dynamically typed languages from statically typed ones. The questions are:

- Do developers ever need/use this behavior?
- Is it natural for developers to view variables as dynamic containers or do they tend to assign types to variables and just not write them down in dynamically typed languages?
- If developers do use this dynamic feature, is it more present in instance variables, method arguments or temporary variables?

To attempt to answer some of these questions, we propose a study of type consistency in a large number of projects written in dynamically typed languages. Analyzing the changes in types of objects that variables hold we can conclude how often, if ever, objects of radically different types are assigned to the same variable. We look at smalltalk as a starting point.

MAIN TASKS:

- Find a way to monitor how types change in variables during execution of smalltalk programs
- Run as many smalltalk programs as possible with the monitoring active
- Store the data collected from the monitoring to a database
- Analyze the collected data and draw conclusions about developer behavior.
- Answer the question from the title of this page.

Project mentor:
Boris Spasojević (http://scg.unibe.ch/staff/Boris-Spasojevic)