David Ungar
Position Statement

For over fifteen years, I have been involved with Self, a dynamic, pure object-oriented language based on prototypes instead of classes. When we designed and built Self, we wanted not to implement a language, but rather to create an artificial reality. In fact, Self designer Randall B. Smith has previously built his Artificial Reality Kit, and I found his viewpoint to be inspiring. I had used APL many years before, and the idea that a programming environment should create an artificial world explained a lot about what was good about APL.

What, exactly, is the difference between language-implementaiton and reality-creation? Harmony, unity, integrity. And from these flow the much-bruited benefits of malleability, power, and productivity. Harmony means that your world has only a few concepts, and each one can work with any other to create a unique blend. Consider how every chord in Western music is built out of only twelve notes, or how the entire physical world can be viewed as a collection of only four forces. In Self, any slot, be it state or behavior, can go into any object. But in Java-Smalltalk, state goes in objects and behavior goes in class objects.

Unity means that the elements of language and environment combine into one whole. Objects are live, code always runs. There is no difference between compile-time and run-time. In fact, we use every trick in our book to conceal compilation. The user makes a change, commits it, and in a fraction of a second, the change takes effect. Some systems, such as Magpie, commit changes after every keystroke. This immediate response mirrors the real world. When you let go of a rock, you don't have to wait for twenty seconds for the Universe to compile the changed conditions. As humans, we are most able when functioning in a world with immediate feedback. Any perceptible delays disturb and distract us.

Integrity means that the system tells a single, consistent, truth. Wild-pointer errors violate integrity because what happens when you scribble over the return PC on the stack has nothing to do with the truth of the language. Class-path error messages violate integrity because they refer to conditions outside of the world of your Java program. In Self, one can grab most any object and change it in anyway, adding, modifying, or removing any slot with either state and behavior. The few exceptions (one cannot add a slot to an integer), weaken Self's artificial reality.

When you build a system that creates an artificial reality, its users easily immerse themselves, the system gets out of the way, and the true creative abilities of your users are let loose to make magic. That's why I believe in dynamic languages.