# The Programmer as Poet

Russell Jay Hendel
Department of Mathematics, Towson University
Towson, MD, 21252, USA
E-mail: RHendel@Towson.edu

## Abstract

In Tennyson's *Now Sleeps the Crimson Petal,* the poet requests from his lover, "...*slip into my bosom and be lost in me."* This theme is poetically developed by seeking an oneness with nature: The poet reviews many natural events which have a cycle of energetic wakefulness followed by a state of relative rest. A similar method of poetic development, by analogy with several other domains, occurs in other poetic passages: for example, Job wishes death by metaphorically seeking that his day of birth be *lost, stained, unlit, not allowed to come to the calendar,...* Computer scientists will immediately recognize this technique of poetic development as resembling **polymorphism**, which allows the naming of an abstract concept by its instantiation in one particular domain. This paper explores use of computer concepts to classify poetic technique; it also advocates enriching computer science curriculum with the teaching of poetic technique.

## 1. Goals

**1.1. Overview.** We first summarize 2 poems from different periods, with distinct styles, authors and topics. Our exploration of a common structure in these 2 poems naturally uses modern computer concepts. This suggests a computer-poetry classification project from which both disciplines can benefit.

**1.2. Now Sleeps the Crimson Petal [1] [2]**. The climax of Tennyson's poem, Now Sleeps the Crimson Petal is a request for the lover, after consummation, to slip into the poet's bosom: *So fold thyself, my dearest, thou, and slip into my bosom and be lost in me.* This theme is poetically developed by seeking an oneness with nature. The poet reviews many natural processes that similarly start in a more energetic state followed by a more restful state: (a) a *meteor* coming to rest, (b) *day* turning to night, (c) *lilies* nightly folding into a lake, etc.

**1.3. The Death Wish, (Job, Old Testament, 3:3-11).** In the dramatic opening passages of the poetic section of the book of Job, the poet wishes death: *Why did I not die from the womb? Why did I not perish when I came out of the belly?* This death wish is poetically developed by applying a variety of non-existence concepts from other domains to the date of birth of the poet. Examples of requested non-existence in the poem include (a) *losing* the date of birth, (b) *staining* it, (c) not letting it *come* to the calendar, (d) *darkening* it, etc.

**1.4. Polymorphism.** To define the commonality of poetic technique in the above passages we briefly review the computer concept of **polymorphism** [3]. A standard example is illustrated with a **class**, *OrchestraInstruments*. Suppose the only instrument you know is the *Violin*, which **extends** the **parent class** *OrchestraInstrument*. You know that violins are *fiddled* but at the time of writing the program you don't know the names of any other *OrchestraInstruments* or how they are *played*. Proper programming technique would write an *OrchestraInstrument* **method**, *fiddle()*. Then, every particular instrument could **inherit** this method but adapt its definition to that particular instrument. Thus, the program would speak about *flute.fiddle(), drum.fiddle()* etc. The word *fiddle* now represents the abstract concept of *playing* an instrument. The programmer finds useful the indefiniteness of the *fiddle* **method** since it can be defined in each new situation as needed [3]. Thus, the programmer has used metaphoric technique to write a program. Modern compilers can compile such a program even though at the time the program is written no knowledge is possessed of other instruments and other playing methods. Computer theory calls this **polymorphism**. **Polymorphism** enables us to articulate the commonality in the Tennyson and Job passages both of which poetically develop a single theme by polymorphically / metaphorically using similar concepts in other domains.

**1.5. The Symmetry Classification Project.** [4] describes a joint classification project by anthropologists and group theorists in which the group of symmetries of the patterns in the clothing, pottery and blankets of several distinct cultures and tribes were analyzed. The resulting analysis surprisingly uncovered mathematical categorizations of symmetries of different cultures. Based on the analysis presented in the preceding sections, we would suggest a similar project between poets and computer scientists: that is, advanced computer science concepts could be used to provide classifications of the types of poetic development employed in the poems of different poets or of different periods. This computer-poetry classification project could uncover mathematical categorizations of poetic development useful to poets.

**1.6. Poetry vs. Mechanism.** In this subsection we discuss the paradox that computer science with its classical emphasis on objectivity, precision and mechanism should be useful to poetry with its romantic emphasis on feeling and freedom from precise laws [5]. A possible resolution to this paradox was presented in section 1.4. There we showed how computer science, within its own mechanistic borders, now sees a need for concepts of an *indefinite* nature. To the poet metaphors create a "mood of musical like *indefiniteness"*[5]; to the programmer concepts that are *indefinite* are useful. It follows that the traditional classical-romantic distinction must be reexamined. Neurological studies have similarly uncovered a need for intelligence tests of higher order rationality not covered in standard intelligence tests[6].

**1.7. Computer Pedagogy Enrichment.** The preceding subsections dealt with computer theory aiding the poet. But the converse approach should also be fruitful. If computer programs are now using poetic concepts then perhaps knowledge of basic poetic technique will enhance the performance of the programmer. Anecdotally, all computer science instructors know that a basic pedagogic problem in introductory courses is an over-emphasis on mechanism, objectivity and correctness. It is equally important to emphasize overall structure and use of analogy. A re-examination of the computer curriculum with an emphasis on the poetic, would therefore be welcome.

## References

[1] Helen Gardner, ed., *The New Oxford Book of English Verse*, Oxford: Oxford UP, pg. 648. 1972

[2] Gerald J. Steen, *Identifying Metaphor in Language: A Cognitive Approach*, Style, Vol. 36.3, pp. 386-407. 2002.

[3] Bruce Eckel, *Thinking in Java, 3rd edition*, Prentice Hall, 2003.

[4] D. K. Washburn and D. W. Crowe, *Symmetries of Culture: Theory and Practice of Plane Pattern Analysis*, Seattle, Washington: University of Washington Press, 1988.

[5] Edmund Wilson, *Axel's Castle*, N.Y.: Charles Scribner's Sons, pp. 1-25. 1959.

[6] Antonio R. Damasio, *Descartes' Error*, N.Y.: Grosset / Putnam Book, 1994.