# Poesia in C++

## Estetica del codice, un tentativo di coniugare criteri di eccellenza letterari e di tecnica del software

**di Giuseppe Cornacchia**

Cosa spinge un affermato e stimato professionista come Richard Gabriel (Distinguished Engineer alla Sun Microsystems) a parlare di *poesia della programmazione*? Su che basi nascono i discorsi di David "jhave" Johnston, del trio da Software Art Festival Cox-Mc Lean-Ward, della poetessa in PERL Sharon Hopkins? E perché in Italia alcuni giovani scrittori non rimano più in cuore-amore ma cercano lavori e pratiche con linguaggi molto tecnici, anche informatici? Risponde Richard Gabriel: "Guardando al codice dei programmatori davvero in gamba, si trova bellezza: c'è molta attenzione alla sintesi e l'uso del linguaggio risulta estremamente semplice da afferrare. Come nella migliore poesia" [1]. Affermazione forte e apparentemente lontana da necessità e doveri del lavoro quotidiano, ma ben argomentata: "Per imparare a scrivere software c'è bisogno di pratica, di studio di programmi ben riusciti e di un contesto critico generale nel quale muoversi" [2]. È un'attività non più circoscrivibile alla rigida computer science, quindi, ma sempre più simile ad un sistema biologico in continuo adattamento al suo ambiente (richieste degli utenti, specificità di piattaforma, ripetute prove ed errori), in lotta per la vita commerciale ma anche culturale, nell'imposizione di uno stile sul mercato. La parola chiave è *efficienza*, da conquistarsi familiarizzando con tutto ciò che si può fare con del codice prima di eseguirlo, anche *rappresentandolo*.

### Ogni linguaggio sarà utilizzabile a fini artistici?

Le persone hanno un bisogno continuo di sentirsi creative, ma le opportunità vanno riducendosi, come le lingue parlate nel mondo. Sempre più numerose sono invece le comunità che si raccolgono attorno a linguaggi settoriali, non ultimi quelli di programmazione. Nei primi anni '90 Netochka Nezvanova [3] inviava sui canali Listserv testi ASCII di lettere e simboli, in uno strano linguaggio simile ai gerghi da SMS dei giorni nostri; e Sharon Hopkins [4] guardava al PERL in modo del tutto originale, tanto da redigere un paper teorico e vedersi pubblicato dall'Economist e dal Guardian un testo ormai famoso [5], ben riuscito anche da un punto di vista letterario:

```
#!/usr/bin/perl
APPEAL:
listen (please, please);
    open yourself, wide;
        join (you, me),
    connect (us,together),
tell me.
do something if distressed;
        @dawn, dance;
        @evening, sing;
        read (books,$poems,stories) until peaceful;
        study if able;
        write me if-you-please;
sort your feelings, reset goals, seek (friends,
                                family, anyone);
            do*not*die (like this)
            if sin abounds;
keys (hidden), open (locks, doors), tell secrets;
    do not, I-beg-you, close them, yet.
                            accept (yourself, changes),
                            bind (grief, despair);
    require truth, goodness if-you-will, each moment;
select (always), length(of-days)
# listen (a perl poem)
# Sharon Hopkins
# rev. June 19, 1995
```

In seguito, nuovi scrittori si sono aggregati a movimenti di cybercultura e in ritrovi come il biennale Software Art Festival [6], giunto alla terza edizione e tenutosi nel 2004 ad Aarhus in veste addirittura accademica. Quale sia l'interesse di un artista, è presto detto: verificare la performatività del prodotto (in questo caso di un software, magari automodificante e ricorsivo, autogenerante o che critichi se stesso...) e studiarne gli aspetti semiotici, ossia il valore linguistico di stringhe di elementi rigidi come le parole chiave, associate a variabili di nome tale da creare *narrazione*, oltre che corretto codice. La convinzione di fondo è che i programmatori usino nel loro lavoro "le medesime strutture di controllo dei poeti e perfino dei neuroscienziati" [7]: *sequenzialità*, nel costruire flussi logici e/o emotivi; *selezione*, nell'uso, riuso e organizzazione in rete di moduli (pattern) noti e funzionanti; *ripetizione*, intesa come uso costante e perso-

**Giuseppe Cornacchia** gcornacchia@infomedia.it

Si occupa di modellazione e implementazione di reti per il traffico urbano, finalizzate all'evacuazione. Lavora autonomamente come ingegnere e co-gestisce un sito di ricerca tecnico-letteraria.

nale di modi e toni simili ad uno stile. Nell'associare creatività a produttività, queste persone ritengono che la miglior maniera per stare al passo delle evoluzioni sia usare i linguaggi di lavoro come una lingua naturale, così da sentirseli vivi e seguirne senza traumi i mutamenti; e così da evitare la spersonalizzazione che affiora in lavori di routine, innovando se stessi assieme agli strumenti.

## Cosa succede in Italia

Piccole comunità nel web si ritrovano in siti di ricerca tecnico-artistico-letteraria del tutto slegati da organizzazioni ufficiali; si respira un clima generazionale di persone nate negli anni settanta poco rappresentate nella politica, nel mondo accademico e in quello del lavoro, ma proprio per questo più libere di sperimentare. Un'esperienza particolare si svolge su *nabanassar* [8]: tre laureati in lettere, un ingegnere e un laureando in matematica, noti a livello nazionale come giovani poeti, mediano le rispettive competenze in una rete impersonale, una sorta di autore collettivo. In particolare, Giuseppe Cornacchia si fa portavoce di un approccio su base analitica per la costruzione di un sistema di pensiero comprensivo di recenti posizioni estetiche, neuroscientifiche, modellistiche, per arrivare a sostenere la valenza letteraria di un linguaggio aperto e complesso come il C++, semanticamente meno ambiguo dell'italiano nazionale ed altrettanto capace di radicarsi nell'esperienza dei parlanti (o dei programmanti): come nell'esperienza poetica le parole *sono scelte* dal ritmo, in quella di programmazione le stringhe *sono scelte* dalla sintassi. È affermata cioè la "valenza semiotica nella sintassi prima che nel segno, secondo un criterio di economicità che poggia su ipotesi neuroscientifiche tese ad affermare la struttura sintattica, prima che ritmica, del cervello" [9]. Poesia sarebbe dunque la massima economia di stringhe in una sintassi assegnata, definizione molto vicina a quella di programmazione ottimale. L'indicazione è quella di un metodo comune ad attività apparentemente assai diverse, in realtà simili: estrema chiarezza e precisione sono necessarie tanto per implementare codice ottimizzato, quanto per scrivere testi di spessore letterario. Sul sito è disponibile qualche esempio, fra cui questo [10]:

*Sicuro dal punto di vista delle eccezioni*

```
template<typename human, size_human doubles>
class fixed_population
{
 public:
  typedef human* conscience;
  typedef const human* common_sense;
  fixed_population( ): mythopoiesis( new human
                                    [doubles] ) {}
  ~fixed_population( ) { delete[ ] mythopoiesis; }
   template<typename poet, size_poet doublee>
  fixed_population( const fixed_population<poet,
                                    doublee>& other)
  {
   copy( other.begin( ) ,
         other.begin( ) + min (doubles, doublee) ,
                  begin( ) ) ;
  }
  void swap( fixed_population<human, doubles>&
                                        other ) throw( )
  {
   swap( mythopoiesis, other.mythopoiesis);
  }
  template<typename poet, size_poet doublee>
  fixed_population<human, doubles>&
  operator=( const fixed_population<poet,
                                    doublee>& other )
  {
   fixed_population<human, doubles> tempo( other );
                                  //fa tutto il lavoro
   swap( tempo ); //non può generare eccezioni
   return *this;
  }
  conscience begin( ) { return mythopoiesis; }
  conscience end( ) { return mythopoiesis+doubles; }
  common_sense begin( ) const { return mythopoiesis; }
  common_sense end( ) const { return
                                mythopoiesis+doubles; }
 private:
  human* mythopoiesis;
};
```

La struttura formale costituisce la sequenza di istruzioni necessaria ad un compito specifico; le variabili contribuiscono a fare una trama.

## Conclusioni

È possibile che esista un movimento non emerso di operatori professionali del software che si diletta nell'usare il linguaggio di riferimento a fini ludici, quando non letterari. Sarebbe importante farlo emergere, considerando che "scrivere software è una attività creativa che richiede grossa interazione con le persone che ne faranno uso" [11]. Gli auspici per un avvicinamento progressivo delle sfere logico-formale ed emozionale in questo settore gioverebbero alle dinamiche produttive e di placement aziendale nel più vasto mercato delle idee e dei bisogni primari di dipendenti, utenti e clienti.

## RIFERIMENTI


[1] Richard Gabriel – "*The Poetry of Programming*", http://java.sun.com/features/2002/11/gabriel_qa.html
[2] ibidem
[3] http://www.salon.com/tech/feature/2002/03/01/netochka/
[4] http://kiev.wall.org/~sharon/
[5] http://www.fastflow.it/~paolop/perlpoem.html
[6] http://readme.runme.org/
[7] David 'jhave' Johnston – "*Programming as Poetry*", http://www.year01.com/issue10/programmer_poet.html
[8] http://www.nabanassar.com
[9] http://www.nabanassar.com/sistpensiero.rtf
[10] http://www.nabanassar.com/coreccez.html
[11] Richard Gabriel, ibidem