Organic Programming

September 13, 2017

Organic Programmation with C+Pie

C+Pie is better understood as a sens and motor extension of the human rather than a GUI. It is based on 'Organic Programming', a possible naming for a paradigm which adds 'immanence' and 'corporality' to OOP. Immanence means, that data and real objects are conceptualized in the same way and without prioritizing the one category over the other. The justification of the term corporality needs a bit abstraction. To a pen and glasses would be associated the words output and input respectively because for this items an anthropocentric stance is taken, a keyboard is an input device because of an computer oriented point of view, and we take something 'out' of a box to put it 'in' an other because the point of view is centered on the object, the box. Common to all, is that they can be seen in an object centered way: a keyboard has as input the hand and as ouput sends a signal to the computer. In C++, the world 'cin' for the representation of the keyboard is inherited from a program centered point of view, and would have been named maybe 'csout', a stream of somewhat open for output in an 'Organic Programming' aware context. Similarly, a file opened for reading would be a 'fout' object. A relevant difference then between 'csout' and 'fout' is that 'fout', like the box, keeps its object centered point of view, while the 'out' in 'csout' can be interpreted in two ways: the 'out' of the keyboard abstraction directed to the program, and the anthropocentric 'out', the keyboard is interpreted like a pen would be. 'Organic Programming' is not to enforce an anthropocentric point of view but to release from the system point of view, while corporality is then the result of the intuitively done distinction between objects which are interpreted anthropocentrically from the others. In accordance with the concept of immanence, objects are code and other things: in 'Organic Programming', programming is understood in its general sens.

In the practice of coding, 'Organic Programming' improves robustness, portability, standards independence, and expressiveness of the code. Beside better maintainability and reusability it encourages restructuring, which could be dubbed 'mutability'.

C+Pie stands for C++ integral environment or C++ plus Pi and e for the exponential number to stress its affinity to mathematics. One of its most unusual aspects, is that it does not differentiate between usual input method to a direct

change of the code. Because the program is distributed between processes on the same or other machines, the linking and relaunching on consumer class machines goes in about 1 second. Secondly, it is a 3D environment. Each non abstract object has a volume and a time space location, which can not be shared with other, and the object are positioned by gravity, their inertia and their elasticity. The graphical representation departs from the invalidation and repainting mechanism of many APIs. In C+Pie, the draw surface, the 'retina', is an 2D array of 'cells', a linked list of color-z pairs, z being the distance of the color particle form the origin of the cell. Together with the position and orientation of the 'retina', the position of the color point is defined. The objects are responsible for the management of the color-z pairs. Because a cell is a linked list, an object does not lose the reference of its color point when a color point is removed or inserted by an other object, and because all color points have a z value, it knows where to insert his. The image is simply the sum of the top inscription of the cells. Thirdly, an organic entity is the triplet user, machine and an instance 'me' of the class 'lamb_c', and all elements of the triplet are separable. The interface between 'me' an the machine goes through the class 'message_c', and this interfacing is dynamically portable. While 'me' is running, one can change the machine even with an other operating system. Nevertheless, breaking the principle of a triplet but to simulate different machines or to run a debugger on a clone of C++Pie, there is an emulation mode were it is possible to run more instances of 'me' on the same machine.

There is no notion of window or window manager, each object inscribes in the retina in its on way, there is no toolkit and a coherent look and feel would as matter of applicability rather be commanded by real objects. The program is highly portable and currently run on the Linux framebuffer and X11. It is projected to run C+Pie directly on the Linux kernel without shell. In the source there is the shell 'ccshell'. A 'ccscript' is an executable which contain its own source inclusively makefile and by derivation those of its libraries. Each 'ccscript' can change from source to executable or reversely and expose in itself all the shell. The replacement of X11 and bash is part of the motivation of C+Pie and the reason for 'ccshell'.

C+Pie is shipped with a text editor for variable font, folding and elastic tabtop. It input is programmable so that everyone can keep the key bindings he is used to. The shortcuts are sequential, for example Ctrl+y+x can be given an other signification than Ctrl+x+y. It has also a pdf viewer as proof by application which makes C+Pie even nicer for Tex or Latex users.

C+Pie is experimental for its own virtue.

A few more things

'Corporality' can also be seen as a phenomenon related to the better connectivity between computers, which become more collector of services produced externally. The computer reduces to an interface between the data network and the human. The data are presented in a ready to impress form to the sense of the human, and its motor is exported in increasing details. Also, 'corporality'

may be connected to latency, fundamentally limited by the ratio distance to light speed, and which implies a physical proximity between items needing a quick response in divide to those using a greater throughput.

C+Pie is a 3D environment using a local coordinate system on top of a global one. The rational for this at first glance uneconomical choice is in forecast of change in the transformation. Astrophysicists would prefer a relativist transformation, architects would be content with a transformation with a linear gravitational field and so on.

In a code management system, a branch would always be particular to a user. It would be organized around a group of 'creator' branches of highly mutable code, which appears rather as a protocol in elaboration, and the others group of branches would be 'consumers' who gear the program towards particular purpose while continuously readjusting to the change of the 'creators'. Inherently, C+Pie is not made for standardization, it is the improvement in semantic and structure which makes the interfacing easier.

'Organic Programming' captures AI by binding the machine to the human.

typesetting LaTeX