Boxer: an Appreciation in Five Inventions

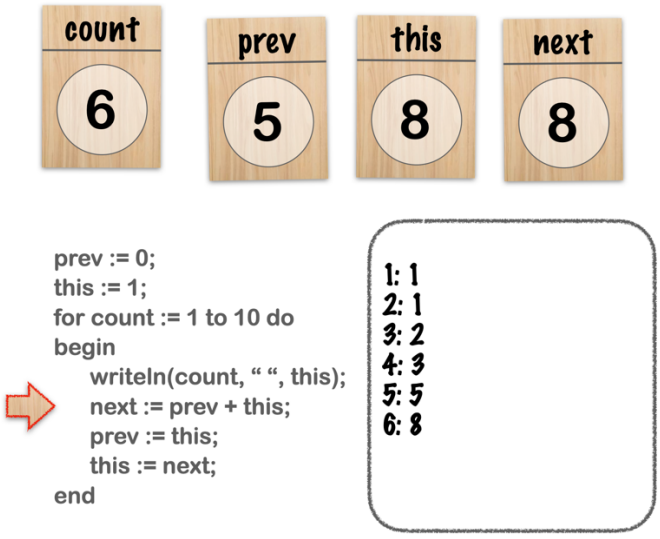
A presentation proposal for the Boxer Salon at <Programming> 2022

Chris Hancock
Tertl Studos LLC
Montpelier VT

The conceit of this talk is to use five different educational technologies as prisms for thinking about key innovations that Boxer brought forth, and a few of the key ideas about learning and design that Andy diSessa has articulated in conjunction with the Boxer oeuvre.

Invention #1: A display set for Pascal

I first learned about Boxer in the mid-1980's, from the edited volume *User Centered System Design: New Perspectives on Human-computer Interaction* (Norman & Draper, 1986). I think I can convey some sense of what a breakthrough this was by comparing an invention of my own from a few years earlier.



As a teacher of introductory programming at Harvard's Extension School, I developed a display set to help students visualize the inner workings of programs. The primary piece was a wooden variable, with magnets on the back, and urethane on the front so that names and values could be written and erased using dry erase markers. Additional pieces addressed data structures, indirect references, and local execution environments.

The display set responded to a dire need in introductory computing education. Having worked with it, I could immediately appreciate some of the power of Boxer. But it is interesting to think about ways that the Boxer environment is more than just an onscreen display set. So I propose to spend a little time with this comparison and contrast.

The Norman & Draper book helped to define an exciting moment in the HCI field. An intellectual frontier was opening up around the virtual realities that could be built with direct manipulation interfaces. Object-oriented programming was also central to that movement. Boxer brought these ideas into the educational realm in a powerful way.

Invention #2: A Boxer-like environment for real-time programming and modeling

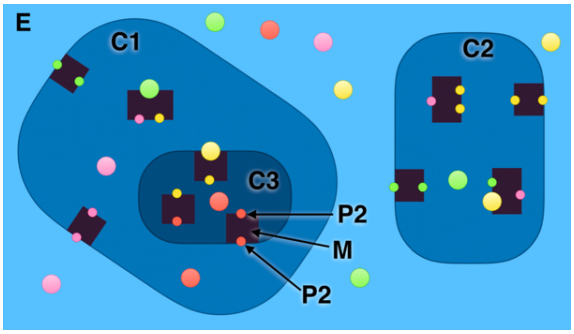
Between 2000-2011 I worked on programming environments for real-time applications such as robotics and game-making, producing the prototype language Flogo II and a broader, more Boxer-like elaboration called the Tertl Environment. The latter work in particular is not well documented and I can't demo it now (a key hard drive was lost in 2013), but I can briefly outline it and highlight a few points of comparison:

- The significance of live program text, integrated declarative programming, and pseudo-continuous time.
- The paradox that of all subjects, Boxer was used perhaps most extensively for work in physics, a highly time-oriented discipline.

- The Tertl Environment featured a small repertoire of built-in tools (e.g. a real time graphing tool) that, in the pure Boxer ethos, ought to have been built within the environment. There are trade-offs both ways.
- Challenges of marketing a Boxer-like environment

Invention #3: A language for ideas of cell metabolism

With smaller, more special-purpose programming languages, Boxer's influence becomes harder to distinguish from the influence of Logo. But I would like to take a crack at it in the context of two language design efforts I have recently undertaken in collaboration with commercial entities.



From boxes to membranes. Midway between a language and a microworld, BubbleFlow is all about interactions and transformations governed by energy constraints. It can enact big ideas of microbial ecosystems and cellular metabolism, such as catalysis, inhibition, paired reactions, energy currency, and optimization of energy flow.

The ports, or receptors, that permit specific kinds of bubbles to enter or leave a cell are very much like subroutine parameters. The ability to encapsulate metabolic functions brings out the analogy between subroutines and mitochondria.

The product envisioned for BubbleFlow would be a more conventional gamified learning product, with puzzle progressions complemented by a free construction mode. How does that compare to the learning modalities supported by Boxer?

Invention #4: Programming blocks for very young children

Learning in (plastic) pieces. The forthcoming television series Codie (© Codie LLC) will feature a block-based language for young children to think about steps, loops, and conditions, and also nascent ideas of literacy that precede the written word. Our notions of literacy can be expanded at this level, as they can at the level of Boxer.

The slogan *programs you can say* captures a key design principle setting this design apart from other block-based languages.

This language was a fascinating and opportunity to try to bring some of Andy's key ideas about learning (learning in pieces, islands of competence, progressive empowerment) to a design for much younger children.

Invention #5: Letterfinger

I'd like to conclude by demonstrating something quite different. Letterfinger is not a programming language, but rather a direct manipulation interface for the written word. It lets children sound out a word with their finger.

Letterfinger's real-time, continuous response promises a critical difference for the many learners thrown off by the standard pedagogy of isolated phonemes that must then be "blended."

The obvious, and powerful, Boxer connection is the value of tightly integrating a notation with its meaning. I will endeavor to draw in some additional principles from the Boxer tradition.