

## **Improving the Usability of Visual Functional Programming Tools**

BOISVERT, Charles <<http://orcid.org/0000-0002-3069-5726>>

Available from Sheffield Hallam University Research Archive (SHURA) at:

<https://shura.shu.ac.uk/29928/>

---

This document is the Published Version [VoR]

### **Citation:**

BOISVERT, Charles (2021). Improving the Usability of Visual Functional Programming Tools. In: PPIG 2021 32nd Annual Workshop, Online, 21 Jun 2021 - 25 Jun 2021. [Conference or Workshop Item]

---

### **Copyright and re-use policy**

See <http://shura.shu.ac.uk/information.html>

# Improving the Usability of Visual Functional Programming Tools

Charles Boisvert

c.boisvert@shu.ac.uk

Department of Computing  
Sheffield Hallam University

## 1. Introduction

This short report presents reflections on the design challenges of a visual functional programming environment. Some early work is presented in (Boisvert, Roast, & Uruchurtu, 2019, 2021); here we will investigate design difficulties and propose reflections on what they reveal.

## 2. Some Background

### 2.1. The *boxes-and-wires* model

Lambda calculus' mapping to directed acyclic graphs provides a visual model, summarised table 1. The graph, or model, can read as a data flow.

Notation	Represents	Graphical equivalent
$x$	Variable	$\longrightarrow$
$f$	Function $f$ (with one parameter)	$\boxed{f}$
$fx$	Application (function $f$ is applied to a variable)	$\longrightarrow \boxed{f}$
$gfx$	Function composition ( $g$ is applied to the result of $fx$ )	$\longrightarrow \boxed{f} \longrightarrow \boxed{g}$

Table 1 – Basic elements to represent function application as a graph

The visual result (outlined fig 1) is clear and popular: Many visual functional tools build on the *boxes-and-wires* model.

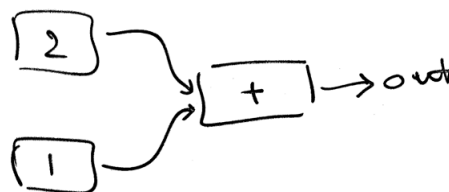


Figure 1 – Outline of a simple workflow in the *boxes-and-wires* model

Current examples include Orange, a data mining tool aimed at scientists (University of Ljubljana Bioinformatics Laboratory, 2019); Unreal, a platform widely used in game development (Unreal Engine Team, n.d.); and Luna, which can be controlled through visual or textual representation (Luna Team, n.d.; Moczurad & Malawski, 2018).

But the complete model has to include  $\lambda$ -abstraction and function application. Both are shown table 2.

Technically, this yields a complete and consistent model. But the difficulty interpreting it is evident: in this representation, functions used as data are wires input into and output from other boxes. This breaks the initially simple model where functions are boxes and data are wires.

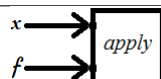

Notation	Represents	Graphical equivalent
$fx$	Application (function $f$ is applied to variable $x$ )	
$\lambda x.f$	$\lambda$ -Abstraction (parameter $x$ is extracted from function $f$ )	

Table 2 – A function for  $\lambda$ -abstraction

## 2.2. Analysing notations

Two frameworks, Cognitive Dimensions of Notations (Green & Blackwell, 1998) and Physics of Notations (Moody, 2009), were used to provide some insight in the notations used. The detailed discussion is available in Boisvert et al. (2021), but some important points are the hard mental operations needed to manipulate  $\lambda$ -abstraction, the very diffuse notation that result when denoting anything but the most trivial processes, and the abstraction-hardness that make it difficult to refactor the graph into a more readable, less detailed form. Figure 2 illustrates this very clearly.

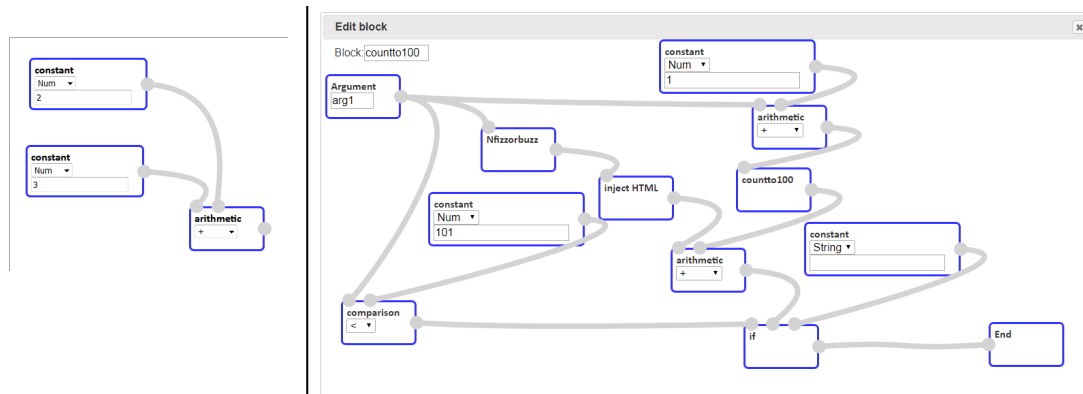


Figure 2 – A simple graph (left) turns unreadable even in a program as simple as fizzbuzz (right).

## 3. Supporting a mental model

One perspective on the problem is provided by Wenger (2014)'s study of Artificial Intelligence and Tutoring Systems. Wenger argues for *epistemic fidelity*: a strong correspondence between the system's internal knowledge representation, its external interface, and the *epistemic source* of the knowledge.

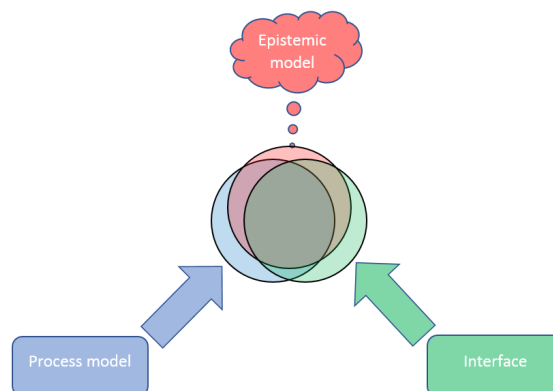


Figure 3 – In a communicable system, epistemic model, interface, and process model closely match

From this point of view, the *boxes-and-wires* model associates a process model to a notation that represents it consistently. But —in its naive form at least— abstraction-hardness makes it difficult to interpret and act on representations, thereby making it a poor epistemic model, difficult to reason with. Improving

interactive representations will give end-users not just an easier to use interface, but means to develop a mental concept through well-chosen representation and interaction.

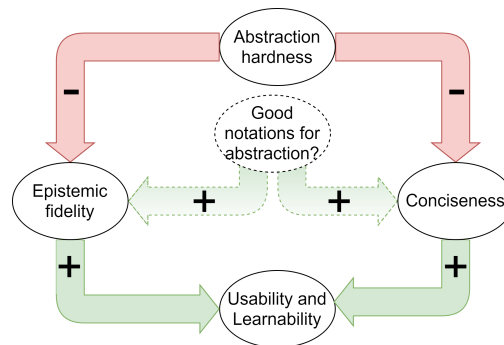


Figure 4 – Abstraction-hardness is the central problem

The work ahead is summarised by figure 4. The difficulty is to find notations that support abstraction, thus allowing the user to navigate between levels of detail and making the visuals more concise, while also supporting and helping form the users’ mental model, that is, in Wenger’s terms, while respecting epistemic fidelity.

We need to represent abstractions that help users navigate complexity:

*How can a graphic representation of functional computation adequately support the varying granularity and levels of abstraction needed to manipulate and mentally process the computation represented?*

And we find suitable representations that more completely align with the user’s understanding as well as a model of computation:

*What complementary notations to the boxes-and-wires diagram support the manipulation and mental processing of needed in functional computation?*

#### 4. References

- Boisvert, C., Roast, C., & Uruchurtu, E. (2019). Open Piping: Towards an Open Visual Workflow Environment. In *Conference proceedings of 2019 International Symposium on End-User Development (IS-EUD)*. Springer.
- Boisvert, C., Roast, C., & Uruchurtu, E. (2021, March). Designing an Open Visual Workflow Environment. Online. Retrieved 2021-05-31, from <https://ppig.org/papers/2020-ppig-31st-boisvert/>
- Green, T., & Blackwell, A. (1998). Cognitive dimensions of information artefacts: a tutorial. In *BCS HCI Conference* (Vol. 98, pp. 1–75).
- Luna Team. (n.d.). *Luna*. Retrieved 2020-12-12, from <https://www.luna-lang.org>
- Moczurad, P., & Malawski, M. (2018). Visual-Textual Framework for Serverless Computation: A Luna Language Approach. In *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)* (pp. 169–174). IEEE Computer Society.
- Moody, D. (2009). The “physics” of notations: toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on software engineering*, 35(6), 756–779. (Publisher: IEEE)
- University of Ljubljana Bioinformatics Laboratory. (2019). *Orange [Computer software]*. Retrieved from <https://orange.biolab.si>
- Unreal Engine Team. (n.d.). *Unreal Engine*. Retrieved 2020-12-12, from <https://www.unrealengine.com>
- Wenger, E. (2014). *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*. Morgan Kaufmann. (Google-Books-ID: 6ymjBQAAQBAJ)