

# Improving Experiences of Computation

Luke Church  
*Computer Laboratory*  
*University of Cambridge, UK*  
[luke@church.name](mailto:luke@church.name)

## 1. Motivation

Experiences of computation vary widely between people. For computer scientists, computation is not only a powerful tool to be used to perform tasks on their behalf, but a fundamental property of nature to be studied, characterised and harnessed.

But others are less fortunate. For many, performing operations on structured information is still a tiresome task [5]. Users ranging from artists [13] to scientists [12] struggle to create programs to meet their needs. Further, the usability implications of designers ‘thinking computationally’ about social relations such as ‘friendship’ are highlighted by problems of privacy in Facebook and similar systems. [11]

This variability in the experience of computation is more than just an annoyance. Many believe [15][6][16][9] that increasing access to computation will help progress other fields. If they are correct, then these problematic experiences represent a barrier to such progress.

My research seeks to address this variability in the experience of computation.

## 2. Research Questions and Methodology

Ultimately, my research seeks to answer the question: ‘*How can we make non-computer scientists’ experience of computation more productive, creative and enjoyable?*’ This is clearly a large question; I currently look at the following aspects:

1. **Computation for Creativity:** How can we use end-user programming perspectives to make consumer interactions with digital photography more engaging?
2. **Computation for Science:** What might biologists do with computation? How can we best support their needs?
3. **Computation for Society:** How can we make security systems more useful for non-computer scientists?

By asking these three questions, I hope to gain a better understanding of how non-computer scientists might use computation to serve their needs. Inevitably there is some variation in the research methodology that I use to address these questions.

### 2.1 Computation for Creativity

Here, the question is one of using computation to generate interesting and engaging experiences. I aim to do this by creating programming environments for end-users that they *enjoy* learning and programming in. One example of such environments is computer games. Games like SimCity are domain specific languages, which end-users successfully and enjoyably engage with.

I seek to characterise the computational properties of such games. Currently I am surveying ‘unconventional models of computation’ such as complex Cellular Automata [16][10].

I will then use such a model of computation to construct an application where a user can ‘execute’ a photograph, resulting in the image being transformed in some way. I will then ask the user to return to the original photograph, modify it in some way and repeat the process resulting in some new behaviour. By observing how the user rationalises the behaviour of system, for which they do not know the computational model, I hope to understand more about the responses of end-users to high complexity computational systems, without the learning costs of conventional syntax.

### 2.2 Computation for Science

Much of the current work in Computational Biology is computer-science driven, typically using some aspect of computer science, such as theorem proving [7], systems theory or theories of random networks [1] to model biological systems.

I act as usability consultant on several such projects, designing languages with which biologists can express their models. This is being done using a typical user centred design approach, with low-fidelity prototypes, think aloud studies and Cognitive Dimensions analyses.

However, some initial results indicate that the computer science driven design processes are not ideal. For example,

in [3] we discussed how communication was possibly a more important goal than analysis. However, communicating the detailed behaviour and design of programs is clearly difficult.

Consequently I am adopting a research approach based around ‘computational toys’, small programs deliberately designed to encourage experimentation [13]. A software developer builds these toys for a scientist to ‘play with’. In the process of this exploration, the scientist is encouraged to consider possible uses, the toy is then refined by the developer and the process repeated, gradually migrating towards a useful tool.

### 2.3 Computation for Society

As noted in the introduction, security systems required for privacy and access control have problematic usability. The body of work in security usability is growing, but much of it is adhoc and guideline driven. My research approach has been to consider security configuration systems from the perspective of end user programming.

I have extracted Cognitive Dimensions activities profiles from seminal works in security usability (e.g. [14][8]), and used these to produce design manoeuvres and more systematic recommendations for the design of security configuration interfaces [2].

However this work has also highlighted a serious problem with the role expressiveness<sub>CD</sub> of the abstractions of security systems for end users.

I am currently working on developing methodologies which are acceptable to the security community for designing abstractions which offer a better fit to end-users’ conceptions of security.

### 3. Initial Results

Many of these projects are still in their infancy, but the initial results seem promising. I have developed a critical perspective for analysing programming environments for their creative potential, partially described in [4] and have proposals for new computational metaphors which I will soon implement.

In the computation for science project we have developed a tool which offers access to powerful computer science functionality (a formal verification engine), and is accessible to biologists with some computational training. Further, I have initial proposals for the first generation of ‘computational toys for scientists’.

My research on security has resulted in a number of profiles for use in design of security systems, which are more concise, complete and systematic than current design guidelines. I also have an early recommendation for using end-user programming techniques for security configuration.

### 4. References

[1] Alon, U. 2006. An Introduction to Systems Biology: Design Principles of Biological Circuits, Chapman & Hall

[2] Church, L. 2008. End User Security - The democratisation of security usability. To appear: Security and Human Behaviour, 2008

[3] Church, L., Apagyi, K., & Fisher, J. 2008. Languages for Biological Models: Importance, Implications and Challenges. Proc. of the 4th Psychology of Programming, Work in Progress Meeting

[4] Church, L. Blackwell, A.F. 2008. Higher Order Design as Shaping Emergence. Proc. of the 4th Psychology of Programming, Work in Progress Meeting

[5] Church, L. & Blackwell, A.F. 2008. Structured Text Modification Using Guided Inference. Submitted to IEEE Symposium on Visual Languages and Human-Centric Computing, 2008.

[6] Emmott, S. & Rison, S. 2006. Towards 2020 Science. Microsoft Research

[7] Fisher J. & Henzinger T.A. 2007. Executable Cell Biology. Nature Biotechnology 25 pp. 1239-1249

[8] Garfinkel, S.L. 2005. Design Principles and Patterns for Computer Systems That Are Simultaneously Secure and Usable. MIT

[9] Hopper, A. 2007. Computing for the Future of the Planet. Presented at The Computer Laboratory, University of Cambridge, 14/02/2007

[10] Inokuchi, S., Honda, K., Lee, H.Y., Sato, T., Mizoguchi, Y. & Kawahara, Y. 2005. On Reversible Cellular Automata with Finite Cell Array. Unconventional Computation, pp. 130-141

[11] Li, Z. 2008. Personal communications, 02/05/2008

[12] Segal, J. 2007. Some Problems of Professional End User Developers. IEEE Symposium on Visual Languages and Human-Centric Computing, 2007. pp. 111-118

[13] Turner, G., Weakley, A., Zhang, Y. & Edmonds, E. 2005. Attuning: A Social and Technical Study of Artist–Programmer Collaborations. Proc. PPIG 17, pp. 106-119

[14] Whitten, A. & Tygar, J. D. 1999. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. Proc. of the 8th USENIX Security Symposium

[15] Wing, J. M. 2006. Computational Thinking. Communications of the ACM, Vol 49, 3, pp. 33-35

[16] Wolfram, S. 2002. A New Kind of Science. Wolfram Media, Inc.