Web Eden: support for computing as construction?

Meurig Beynon
Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK
wmb@dcs.warwick.ac.uk

Richard Myers
RJM Solutions
Haverflatt, Burrells
Appleby CA16 6EG, UK
Richard@rjmsolutions.com

Antony Harfield
Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK
ant@dcs.warwick.ac.uk

ABSTRACT

As Ben-Ari has observed, whatever the merits of adopting a constructivist pedagogical stance towards Computer Science education (CSE), it is impossible to reconcile the classical view of computer science with a constructivist epistemology. There are nonetheless good reasons for wishing to invoke a broader epistemological framework in connection with modern developments in computing practice. These include: the extent to which computing technologies must be studied in the broader engineering context; the greater prominence that the experiential and phenomenological aspects of interaction with computers have acquired; the aspiration (e.g. in agile methodologies) to construct computer artefacts as an integral part of gaining the domain knowledge required for complex software development. This paper proposes Empirical Modelling (EM) as a constructivist pedagogical approach that promises to address such broader issues in CSE within a constructivist epistemological framework. In the light of Ben-Ari's insights, this is possible only through adopting an alternative view of the nature of computing. The Web Eden interpreter is introduced as a suitable first prototype for an EM tool to support this vision for "computing as construction".

Categories and Subject Descriptors

K.3.1 [**Computer Uses in Education**], K.3.2 [**Computer and Information Science Education**]: Computer Science Education, D.2.6 [**Programming Environments**]: Interactive environments.

General Terms

Design, Experimentation, Human Factors, Languages, Theory.

Keywords

Computer Science Education, educational technology, epistemology, constructivism, Empirical Modelling.

1. CONSTRUCTIVISM AND COMPUTING

1.1 Issues for Computer Science Education

The educational emphasis of classical computer science reflects the perception of the computer as a reliable, predictable device suitable for performing computation in the sense identified by Lynn Stein [2]: "Computation is a function from its inputs to its output. It is made up of a sequence of functional steps that produce – at its end – some result that is its goal." Teaching programming stands at the core of the classical discipline. Learning to program involves using formal programming languages whose syntax and semantics is not negotiable. As Mordecai Ben-Ari observes [1], whilst CSE that respects this tradition may benefit from a constructivist pedagogical stance, it cannot embrace a constructivist epistemology such as has been the focus of controversy in the philosophy of science (cf. Latour [3]).

Modern computing nonetheless provokes questions that are not easily addressed by traditional computer science. For instance:

a. How should we place classical Computer Science in the broader engineering context? Applying computing technology in complex systems raises concerns similar to those traditionally associated with engineering. In asking "What can we expect of formal verification?", the distinguished software consultant Michael Jackson highlights the need to take fuller account of the engineering perspective in complex systems development. And whilst Ben-Ari remarks upon the affinity between CSE and engineering education [1], he identifies the extent to which his findings generalise to engineering education as an open question.

b. To what extent are experiential and phenomenological concerns within the scope of Computer Science? In many modern applications of computing technology, the primary emphasis in requirements is on experience rather than abstract function. In such applications, the concrete physical characteristics of the technology itself play an essential role. We may reflect on what considerations affect the merits of different devices and interfaces for speed-texting, for instance. This leads us to think of the computer as resembling an instrument and to recognise the impact that acquired skills have upon effective performance.

c. *Can we interpret programming activity as a legitimate way of developing domain understanding?* Agile methodologies feature prominently in contemporary software development. In such approaches – contrary to the precepts of traditional programming – the conception of the software product and the understanding of the domain this presumes are apparently being acquired even as the product itself is being constructed. Interpreting a piece of software as embodying domain and problem understanding rather than merely meeting a functional requirement raises challenging philosophical and ontological questions (cf. Loomes and Jones [4]). The Play-In approach to software development advocated by David Harel illustrates a process of software construction that resembles the negotiation of meaning in a constructivist idiom.

Such questions all relate to how far we can conceive interaction with computers as "computational" in the narrow sense of Stein [2]. Accepting that interaction with computers must be program-like in this sense makes it hard even to *formulate* these questions. This has motivated many critiques of classical computer science.

1.2 Broader Visions of Computer Science

The discrepancies between computing practice and what classical computer science addresses have been noted by many researchers and interpreted in many different ways. Writing in 1998, Ben-Ari [1] remarked that "the gap between the standard libraries (especially the GUI libraries) of a modern programming environment and the model of the computer is so great that motivating beginners has become a serious problem". For Ben-Ari, the GUI libraries are obstacles to the appreciation of the computer as an "accessible ontological reality" of which the student must develop a mental model. By contrast, Winograd and Flores [5:78] contend that "computers do not exist, in the sense of things that possess objective features and functions, outside of language" and argue for a reconceptualisation of computing beyond the "rationalistic" epistemological framework. Ridley [6] articulates the perplexing issues that surround database theory, where the relational model that was once viewed as the foundational cornerstone of the field is widely perceived as inadequate to account for modern practice.

The fact that Margaret Boden [7:1414] reviews the history of the concept of computation under the heading "Computation as a Moving Target" reflects the subtlety of the notion. Brian Cantwell-Smith [8] highlights the inadequacy of traditional accounts of computation in respect of modern computing practice, and draws particular attention to the fact that what is understood by the "semantics of computation" in theoretical computer science is not to be confused with "the [content relation] that holds between the computational process and the world outside it" (which Smith describes as "the semantics of the semantics of the process"). Stein [2] argues for the need to move from the classical interpretation of "computation as calculation" to "something one might call computation as interaction".

These diverse critiques of classical computer science indicate that there is considerable interest in broadening the scope of the science of computing to embrace issues that cannot be addressed by focusing solely on the classical theory of computation. Ben-Ari [1] offers cogent reasons for believing that computer science as narrowly interpreted as the study of program-like interactions with computers cannot be based on an epistemological framework that embraces a constructivist stance. But whilst the critiques by Winograd and Flores, Cantwell-Smith, and Stein offer helpful insight into what an alternative science and an alternative epistemological framework might be like, they are ill-developed in respect of principles and tools, especially when viewed alongside Turing's profound mathematically-based contribution to our understanding of algorithmic processes.

1.3 Empirical Modelling

The approach to computing to which the Web Eden tool to be introduced in the second section of the paper relates is that of *Empirical Modelling* (EM) [9]. EM is based upon an unconventional epistemological framework that is consonant with William James's radical empiricist philosophical stance [10]. James's conception of knowing is rooted in direct experience – his primary thesis is that relationships between experiences are themselves given in experience. This is the basis on which one experience (e.g. managing one's expenses) can serve as the content of another (e.g. manipulating a spreadsheet). Though such knowing is of its essence a personal matter, this is no obstacle to its potential classification as having an objective quality, if indeed one's own experience is experienced as cohering with that of another person experiencing the same situation (cf. the way in which a financial spreadsheet can represent public information about a company's finances). The nuances to which such a concept of knowing can be adapted are sufficient to admit the kind of realist conception of a computer that Ben-Ari endorses [1], subject to certain reasonable contextual assumptions. It makes good sense to view a computer in this way when considering it as a computational device in a narrow sense for instance, but is not so appropriate if the experience of the computer that is the subject of concern is the colour of the display, or the possibility of erratic operation due to hardware failure is taken into account.

The basic thesis of EM is that there are fundamental and generic principles that can help in constructing artefacts that are intended to be experienced as having a specific content. The key to this construction is introducing counterparts in the artifact for the relevant observables of its referent, and defining dependency relationships – automatically maintained as in a spreadsheet – to reflect the way in which changes to sets of observables are linked in latent atomic changes of state. In EM, the role of such artefacts – known as *construals* – is to mediate the modeller's experiential understanding of a situation before this can be articulated in propositional terms. Developing such construals is conceptually prior to programming activity. Like spreadsheets, construals primarily relate to the representation of a current state of affairs or situation rather than to a process.

EM engages directly with questions *a*, *b* and *c* above.

Because of the fundamental role it gives to personal experience, it is clearly intimately linked with *b*. The way in which EM invokes experiential and phenomenological concerns is well-oriented to an engineering perspective. EM principles can be applied to making sense of situations from the perspectives of human agents with different perceptions and capabilities. By imaginative projection ("to what observables subject to which dependencies can a thermostat respond, and which can it change?"), EM can be applied to other kinds of agent. Building construals is an activity that then discloses viable physical and interpretative mechanisms that might be exploited in applications. In this way, it lays the foundation for many different potential functional uses.

Conventional programming activity and the concerns of classical computer science can be interpreted as a specialised form of interaction within the broader framework that EM affords. The prominence that classical CSE gives to abstraction and logic is a reflection of the fact that the empirical activities associated with the identification of the computer as "an accessible ontological reality" [1] are a matter of prior engineering to be taken for granted. In contrast, EM addresses contexts where the nature and robustness of the would-be computational mechanisms is yet to be established [9:#087]. Such a reconceptualisation of computing enables the blending of engineering and classical computer science outlooks sought in *a*.

The radical nature of this reconceptualisation is highlighted by the insights that EM brings to question *c*. James's epistemological stance maintains that all knowing is ultimately rooted in connections that can be experienced. In EM, building construals is about relating knowing to its experiential roots. Though EM can lead to the realisation of program-like behaviours, this realization takes the form of an enactment of pre-rehearsed interactions within a constructed concrete live environment, rather than the specification of an abstract computational process optimised to a specific pre-conceived functional objective. On this basis, EM is an activity that supports the development of domain understanding, but not an activity that can be properly viewed as programming. And where conventional CSE principles and tools are concerned with situations and interpretations that have been reliably pre-established and with associated knowledge that can be expressed in propositional form, the emphasis in EM is upon principles and tools that support the experimental learning activities that must precede such an understanding [9:#098]. It is for this reason that the principal EM tool, the EDEN interpreter to be introduced in section 2, is of its essence a technology to support learning without reference to any specific domain.

1.4 EM in relation to other critiques

There are many points of contact between EM and the various critiques cited above. In EM, the primary emphasis in interpreting interactions with computers is upon "the semantics of the semantics" in the sense of Smith [8]. There is scope for the negotiation of meaning that is relevant in particular to the social processes that frame the protocols for computer use and the identification of patterns of interaction and interpretation with devices that can be deemed to be program-like. As in Stein's conception of computation-as-interaction [2], much importance is attached to maintaining models of the external current state to which the computing activity refers (cf. for instance Stein's discussion of her use of "bootstrapping directly from physical interaction" to equip a robot with a capacity to read maps [2:19]). The realisation of system-like behaviours through the rehearsal and orchestration of primitive interactions amongst agents is well-aligned with the computational metaphor of "a community of interacting entities" proposed by Stein [2:9].

The crucial difference between EM and the proposals associated with the critiques mentioned above is that the development of EM has been intimately connected with identifying principles and building tools to support their application. These principles are more discriminating in the kinds of analysis and application that they endorse. For instance, in keeping with Ben-Ari's realist view of the nature of the computer [1], they legitimise Winograd and Flores's contention that "[computers] are created in the conversations human beings engage in when they cope with and anticipate breakdown" only in particular contexts. They likewise echo Ben-Ari's reservations about the scope for bricolage in conventional programming by calling into question Turkle and Papert's claims – cited by Stein [2:16] to support her concept of computation-as-interaction – about the amenability of traditional programs to experimental development [11]. And, because they focus upon "the semantics of the semantics" of a computational process rather than its abstract denotational/operational semantics, they challenge the notion that the "new generation of software engineering and design tools" identified by Stein in [2:16] illustrates a decisive shift from the usual computational metaphor.

2. THE WEB EDEN ENVIRONMENT

The Web Eden environment [12] is an online environment for constructing interactive models using EM principles. It represents a radical new concept in technology-enhanced learning (TEL) that has been applied in particular to CSE [9:#107], but – as motivated above – can address any learning domain. By exploiting non-standard principles based on modelling dependency relationships for software construction, it introduces a new paradigm for open source development that blends with the learning experience. Because of its distinctive approach to software construction, Web Eden affords an unusually intimate blending of domain learning with model-building in the spirit of Latour's construction [3, 9:#100]. This gives unprecedented scope for exploiting the environment to support learning in many different idioms. We can use Web Eden to guide learners through traditional tutorial-like learning material. Web Eden also enables the learner to explore live dynamic artefacts (as opposed to static pages of learning material). If the learners become really advanced, they are able to build their own artefacts and associated learning activities. Web Eden can run as a stand-alone environment, or we can embed it inside a virtual learning environment such as Moodle [9:#106].

Web Eden, like a spreadsheet environment, features counterparts of meaningful variable quantities ("observables"), defined connections between these which express the ways in which changing the value of one observable directly affects the value of another ("dependencies") and specific instances of redefinition of observables, both manual and automated, that correspond to meaningful action on the part of different agents. The use of dependency is a common – if implicit – feature of much educational software (e.g. tools like Mathematica, The Geometer's Sketchpad, AgentSheets and Matlab, and learning artefacts such as Cabri Geometry and Logotron's Visual Fractions), and its merits are endorsed by the wide range of educational applications for spreadsheets [13]. The motivating idea that makes Web Eden distinctive is that these merits cannot be fully realised within a conventional conceptual framework for computing [9:#096]. In particular, dependency cannot be integrated into an educational tool based on orthodox software principles (such as Imagine Logo) without compromising its conceptual integrity [9:#104].

Conventional TEL software offers little support for integrating the roles of the teacher (a pedagogical expert who conceives and specifies the educational content, interfaces, learning outcomes and exercises), the learner (typically a naive computer user who interacts with the learning environment through a preconceived interface) and the developer (an expert programmer who implements the environment). The Web Eden environment is open for interaction in all three roles at all times [9:#080]. What is more, the interaction takes essentially the same form for teacher, learners and developers alike. Every change to the current state to the current environment, no matter how it is to be interpreted (for instance, whether it is a change to the specification of the environment, a step in the learning process, or a revision to the interface or the underlying program), can be expressed as a redefinition of observables in the model. All restrictions upon interaction and interpretation are then of their essence purely discretionary, according to the expertise and interests associated with each specific role. This does not preclude the specification of interfaces to constrain the ways in which particular agents can redefine observables where this is appropriate.

Web Eden is a web-enabled version of the EDEN interpreter [9:#106]. EDEN was web-enabled by Richard Myers in a prize-winning final year computer science project at the University of Warwick in 2007-8. It exploits state-of-the-art tools that make it possible for server and client machines to share the computational load in interpreting a model. It also overcomes the problems of efficiently interpreting many EDEN models concurrently by enabling distributed processing and load-balancing over many EDEN virtual machines. Many hundreds of models have been built using EDEN [14]. All such model-building has a strong ingredient of domain learning. Many models have an explicit educational objective and the range of learning applications is broad. Web Eden inherits the qualities of EDEN as an educational technology (cf. the “Applications Area” hyperlink at [9]), creating a platform for the full realisation of the pedagogical advantages for which previous experience of EDEN has offered proof-of-concept, and helping to overcome the practical obstacles to wider dissemination and adoption. It addresses the portability issues encountered in downloading the interpreter and models, simplifies the integration of the EDEN engine with other applications through the use of web interfaces, and is designed to incorporate session-sharing features that obviate the need to set up networks for collaborative and distributed modes of interaction.

The most comprehensive practical introduction to Web Eden and the modelling principles on which it is based can be found in the workshops prepared in conjunction with *The Sudoku Experience* - an online activity for gifted and talented pupils organised by the University of Warwick in July 2008 [12]. In these workshops, novice learners are first acquainted with the basic concepts and techniques that are required for model-building. This involves introspecting about the kinds of observables and dependencies that are significant in solving a Sudoku puzzle. They are then shown how these can be related to other tasks, such as devising formulae to convert between different ways of indexing the squares of a Sudoku grid. Once the principles of model-building have been introduced, their application to Sudoku solution is illustrated with reference to a "colour Sudoku" extension and the automation of a technique that is first conceived and implemented as a 'manually executed' pattern of interaction. In the final workshop, the Web Eden environment is configured to allow collaborative concurrent solution of Sudoku puzzles.

Web Eden was also applied in an online database module in the Virtual Studies in Computer Science (ViSCoS) programme at Joensuu University, Finland in 2008-9. This involved integrating Web Eden with the Moodle environment [9:#106]. In the module, design flaws in the international standard RDB language SQL are exposed by contrasting and critiquing different strategies for implementing SQL over a pure relational algebra notation. This practical and interactive approach to highlighting abstract design issues exploits the scope for open-ended interaction that Web Eden affords, which encompasses the capacity for implementing additional notations within the Web Eden environment on-the-fly.

The Web Eden Sudoku model was re-used in a second-year undergraduate module in December 2008. The Alloy tool for formal specification was used to generate the five essentially different abstract mathematical groups of order 8. To make the structure of these groups more accessible, the 9-by-9 grid in the colour Sudoku model was adapted for displaying and manipulating the corresponding group tables [12]. Simple patterns of redefinition and renaming of elements served to acquaint students without specialist mathematical knowledge with the character of a mathematician's intuitive, rather than purely abstract and axiomatic, understanding of group structure.

Other illustrative examples of the use of Web Eden can be accessed via [12]. The environment has recently been further developed to support more sophisticated online use with personal and public project data. The fact that the essential interaction with online models is mediated entirely through definition of observables prepares the ground for several significant extensions. These include: comprehensive monitoring of interactions that enables intermediate states to be recorded and revisited as if "live"; novel possibilities for collaboration primarily mediated through interaction with artefacts rather than communication based on language; potential for graphical user interfaces for fabricating scripts from templates. And though we have gathered informal evidence in support of our claims [9:090], we recognize the need for more rigorous evaluations through empirical studies.

We envisage the deployment of Web Eden not as the release of a product that meets a clearly preconceived specification, but as initiating an ongoing organic process of continuing development associated with the progressive extension, refinement and adaptation of existing models and of the environment itself to better meet educational goals. Teachers, developers and learners will all participate in this process. A major concern in TEL has been that of standardisation. In 2002-4, the principles underlying Web Eden were effectively deployed at the BBC R&D Laboratories in resolving critical issues of cross-platform portability of digital content. This gives us confidence that, appropriately deployed, Web Eden can offer rich experiences customised to diverse learners and contexts. To achieve this goal, we aspire to bring together representatives from schools, universities and industry worldwide to establish an online “Centre for Constructivist Computing” to promote the creation of models, teaching and learning strategies, and extensions and refinements of the modelling tool through open source development.

3. REFERENCES

[1] M. Ben-Ari. Constructivism in computer science education. SIGCSE Bulletin, 30(1):257--261, 1998.

[2] L.A.Stein, Challenging the Computational Metaphor, Cybernetics and Systems 30(6), September 1999, 1-35.

[3] B.Latour, The Promises of Constructivism, In Ihde, D. (ed.) *Chasing Technoscience: Matrix of Materiality*, 2006, 27-46.

[4] M.J.Loomes and S.V.Jones, Requirements Engineering: A Perspective Through Theory Building, Proc. ICRE'98, 1998, 100-107.

[5] T.Winograd and F.Flores, *Understanding Computers and Cognition*, Addison-Wesley, 1987

[6] M.J.Ridley, Database Systems or Database Theory, Proc. LTSN-ICS TLAD Workshop, Coventry, UK, 2003.

[7] M.Boden, *Mind as Machine: A History of Cognitive Science*, Volume 2, Clarendon Press, Oxford, 2006.

[8] B.Cantwell-Smith, Two Lessons of Logic, *Comput. Intell.* **3**, 214-218, 1987.

[9] Empirical Modelling website and EM papers as indexed at http://www.dcs.warwick.ac.uk/modelling

[10] W.James, Essays in Radical Empiricism, Bison Books, 1996.

[11] S.Turkle and S.Papert. Epistemological Pluralism: Styles and Voices within the Computer Culture, Journal of Women in Culture and Society 16(1): 128-157, 1990.

[12] http://www.warwick.ac.uk/go/webeden

[13] J.E.Baker, S.J.Sugden, Spreadsheets in Education: The First 25 Years, Spreadsheets in Education, 2003.

[14] http://empublic.dcs.warwick.ac.uk/projects/