Engineering Emergence

Susan Stepney, Fiona A. C. Polack, Heather R. Turner
Department of Computer Science, University of York, Heslington, York, YO10 5DD, UK
[susan,fiona,turner]@cs.york.ac.uk

Abstract

We explore various definitions and characteristics of emergence, how we might recognise and measure emergence, and how we might engineer emergent systems. We discuss the TUNA ("Theory Underpinning Nanotech Assemblers") project, which is investigating emergent engineering in the context of molecular nanotechnology, and use the TUNA case study to explore an architecture suitable for emergent complex systems.

1. Introduction

Many large engineering systems are *complicated*: they are difficult to understand, to analyse, and to design, because they cannot easily be separated into simpler parts. They are irreducible, not expressible in terms of properties of their parts alone.

The word *complex* is a synonym for *complicated*. It is also used in a more technical sense, to refer to systems with a relatively small variety of relatively homogeneous agents acting together, obeying simple local rules of interaction and communication, with no global or centralised control, resulting in *emergent properties*. The interactions are relatively simple; the agents could be complicated systems in their own right (such as individual craft in a flock of unmanned aerial vehicles). Despite being made of relatively homogeneous agents, complex systems are also irreducible, because emergent global system properties result from the agents acting together in a (self-)organised manner.

Emergent properties are of interest because, in biological systems at least, they seem to be intimately connected with desirable properties such as fault tolerance, robustness, and adaptability. This has generated much excitement, even hype, with more or less wild claims that all we need to do to get these properties is allow them to emerge, by stripping away the centralised control. Unfortunately, this is more a recipe for anarchy than for emergence. Biological emergence has arisen as the result of millions of years of evolution. If we are to reap similar benefits in our artificial systems, we need to learn how to *engineer* emergence: engineer in desired emergent properties, and engineer out undesired ones.

In this position paper, we explore various definitions and characteristics of emergence (section 2), how we might recognise and measure emergence (section 3), and how we might engineer emergent systems (section 4). We conclude with a discussion of the TUNA project, which is investigating emergent engineering in the context of molecular nanotechnology, and use the TUNA case study to explore an architecture suitable for emergent complex systems (section 5).

2. Complexity and emergence

2.1. Students of emergence

There is a vast literature on the subject of emergence, which tends to fall into three camps. The philosophers of mind are interested in how mind emerges from the physical brain, how intelligence emerges from unintelligent matter. Biologists (or, more usually, philosophers of biology) are interested in how life emerges from inanimate matter. Computer scientists (mainly from the ALife community) are interested in how properties analogous to mind or life might emerge from non-biological substrates, computers. There are also other, smaller, groups interested in different levels of emergence: for example, how solid state physics, or chemistry, emerges from the underlying fundamental laws of physics.

Our interest here is molecular nanotechnology: how macroscopic artefacts might emerge from the actions of billions of molecular scale constructor robots obeying relatively simple construction rules (in a way possibly analogous to how macroscopic organisms emerge, or grow, from single cells, with proteins and other macromolecules performing the constructions).

2.2. More than the sum

Aristotle [6] has one of the earliest definitions of what we now call emergent systems: "things which have several parts and in which the totality is not, as it were, a mere heap, but the whole is something beside the parts", now commonly phrased as *the whole is more than the sum of its parts*.

The totality is not some mere heap of components; it is a different kind of property. This goes against the strong reductionist programme, the search for a 'theory of everything' expressed at the level of fundamental particle physics, from which all other physical laws can be derived as consequences. Anderson, a solid state physicist, famously titles his paper "More is different" [5], and goes on to say: "The constructionist hypothesis breaks down when confronted with the twin difficulties of scale and complexity. The behavior of large and complex aggregates of elementary particles, it turns out, is not to be understood in terms of a simple extrapolation of the properties of a few particles. Instead, at each new level of complexity entirely new properties appear, and the understanding of the new behaviors requires research which I think is as fundamental in its nature as any other."

As we shall see, it is not just the constituent parts, but also the organisation of those parts, the patterns that are formed, the boundary and initial conditions, and the context, that are intimately interwoven components of an emergent system.

2.3. Surprise!

Emergent properties are not simple consequences of the underlying laws; they are something new, something novel. Many authors say, something surprising. For example Ronald *et al* [39] say "The language of design L_1 and the language of observation L_2 are distinct, and the causal link between the elementary interactions programmed in L_1 and the behaviors observed in L_2 is *non-obvious* to the observer—who therefore experiences surprise."[1]

This is too subjective a requirement. A scientific property of emergence cannot depend on the ignorance of an observer. Conway's well-known Game of Life (GoL) [26, 12] cellular automaton (CA) can exhibit moving patterns known as gliders. The 'surprise' definition would have us believe that gliders are emergent only the first time we see them (and not even then, if we have been forewarned!). Whether or not a glider is emergent must be independent of our ignorance. Other authors agree. Abbott [2] states that "an observer's surprise or lack of surprise should have nothing to do with how we should understand a phenomenon of nature", and Clark [17, p.112] explains why emergence "does not depend on the vagaries of individual expectations about system behaviour."

[1] Any emphasis in text quoted in this paper is that of the original author.

Instead of surprise, we can require *novelty*, which need not be defined in terms of the eye of the beholder. Crutchfield [18] explains "Emergence is generally understood to be a process that leads to the appearance of structure not directly described by the defining constraints and instantaneous forces that control a system. Over time 'something new' appears at scales not directly specified by the equations of motion."

Indeed, such novelty need not be defined in terms of even the *presence* of a beholder. This also allows emergence on spatial and temporal scales inaccesible to direct observation. Crutchfield [18] explicitly requires the novelty to be independent of any external observer's expectations: "the patterns that emerge are important *within* the system. ...[they] take on their 'newness' with respect to other structures in the underlying system." He defines *intrinsic emergence*, in which the system itself, not some external observer, somehow notices and uses the novel emergent properties: "the system itself capitalizes on patterns that appear".

Bickhard [13] requires something quite strong: "novel causal powers coming into being at specific levels of ontology". This *downward causation* is discussed further in section 2.6.

2.4. Processes, not things

But how could anything *new* actually emerge from what is already there?

Bickhard [13] argues that this question is ill-posed. It makes sense only if we have a *substance metaphysics* (based on static particles, in which no new substances can emerge, only combinations or blends), but that the world is actually based on *processes* and *patterns*. He argues that there are no particles at *any* level of physical reality: "What we normally consider as constituents, as particles or entities, are persistences of instances of organizations ...they are topological". This process view, as opposed to a particle view, is particularly important in open far-from-equilibrium systems (systems with a constant flow of matter, energy, or information through them): "A vortex in a flow cannot exist if the flow itself does not exist." Hence "entities are stable instances of organizations of underlying processes". Here, new (kinds of) processes and patterns clearly could be possible, could emerge. Bickhard points out that we do not currently have an acceptable *process metaphysics*.

Campbell & Bickhard [16] begin to explore what such a new metaphysics might look like, in terms of process and organisation, and examine process stability in near-equilibrium and (necessarily open) far-from-equilibrium cases. They examine a series of far-from-equilibrium processes with increasing causal emergent powers: ones that need external maintenance by the environment, ones that (tend to) maintain themselves in a given environment (are *robust*), and ones that maintain their ability to maintain themselves in different environments (are *adaptable*).

Abbott [2] discusses a range of far-from-equilibrium dissipative emergent *entities*, and how they cycle energy and material through themselves. He also discusses (an admittedly imperfect, but nevertheless suggestive) software analogy to near-equilibrium entities (the analogy being objects without their own threads of control) and far-from-equilibrium entities (the analogy being agents, processes, with their own threads of control), and why we need to include ideas from thermodynamic computing to understand software entities.

2.5. Levels and timescales

A concept that recurs in discussions of emergence is that of *levels*. The constituent parts are at some lower (or local) level, and the emergent properties are at some higher (or global) level. For example, Emmeche *et al* [22] describe "the idea that there are properties at a certain level of organization which can not be predicted from the properties found at lower levels", and Hordijk *et al* [30] talk of "dynamical systems in which the interaction of simple components with local information storage and communication gives rise to coordinated global information processing".

The emergent properties form higher level structures (of patterns, of agents) in space and time. These higher level agents have their own structure and dynamics, their own (longer) length- and timescales. Longer timescales allow relative stability of higher level patterns. Burns *et al* [14] develop the concept of *timebands* to formalise this notion of different timescales being relevant at different levels of system definition: "The slower [higher level] band (A) can be taken to be unchanging (constant) for most issues of concern to B ... At the other extreme, behaviours in [the lower] band C are assumed to be instantaneous". Bickhard [13] notes that, in physical systems at least, "successively higher levels often require successively lower temperatures to emerge ... each level 'condenses' out of lower levels". Lower temperatures, where things happen more slowly, are compatible with longer timescales.

Still higher level patterns can then emerge from this emergent high-level dynamical structure, to give hierarchical emergence. For example, molecules emerge from atoms; cells emerge from molecules; organisms emerge from cells; ecologies emerge from organisms. In Conway's GoL, gliders and other emergent patterns can be organised to construct higher level structures, such as Turing Machines [37] (an example of engineered emergence).

2.6. Downward causation

One of the most contentious concepts in the philosophy of emergence is that of *downward causation*, as it seems to violate the strong reductionism programme.

Campbell [15] first introduces the term, in the context of evolved biological systems: "Where natural selection operates through life and death at a higher level of organisation, the laws of the higher-level selective system determine in part the distribution of lower-level events and substances." He notes that the "'causation' is downward only if substantial extents of time, covering several reproductive generations, are lumped as one instant for the purposes of analysis" because it is a form of causation "by a selective system which edits the products of direct physical causation". This idea of a higher level concept being defined over 'substantial extents of time' fits in well with Burns *et al*'s [14] definition of lower level durations being collapsed into higher level atomic events.

Faith [23] also argues against strong reductionism, except in certain very specific circumstances: "it is only accurate to say that properties of parts determine those of wholes when the entire system is in a narrow range of thermal equilibria. Outside of these specific cases it is equally true to say that *the properties of the parts are determined by those of the whole*", because "every object is ... *situated* in an overall context, and ... it will only have the properties it does because of that context. The causal dependence between parts and wholes goes down, as well as up."

Some authors include downward causation as a defining property of emergence. O'Conner [33] (as quoted by Bedau [10]) defines emergence as: "Property P is an emergent property of ... O iff ... P has a direct ('downward') determinative influence on the pattern of behaviour involving O's parts." Bedau [10] himself characterises this requirement for downward causation to be *strong* emergence, and allows weaker definitions of emergence, since he has problems with downward causation: "Such causal powers would be quite unlike anything within our scientific ken".

Abbott [2] also rules out downward causation, claiming that all emergent properties are "causally powerless". However, he allows that emergent properties are "downward entailing", that is, properties of the emergent phenomenon can be proved to require certain properties of the lower level implementing system, a weaker property than downward causation, but crucially important to his argument.

Bickhard [13], however, argues that downward causation is a necessary criterion for (interesting) emergence, and gives several examples of it in addition to the effect of natural selection on organisms (Campbell's [15] motivating example). Campbell & Bickhard [16] state "That downward causation occurs is a fact".

These arguments are philosophically very important.

From an engineering perspective, however, this careful distinction between downward causation and downward entailment is probably less significant. Engineering is about carefully arranging low level components and processes in such a way that the required high level system properties are exhibited. Whether or not the realised system causes, or merely entails, the lower level implementation, is relatively unimportant to the engineer. What is important are the design processes by which the low level parts can be assembled into correct, dependable and safe implementations of the required system. We need new such processes for designing emergent properties of complex systems.

2.7. Dynamics and attractors

Many authors draw a connection between emergent properties and dynamical system attractors, and particularly strange attractors. (Beer [11] provides a concise overview of the relevant dynamical systems theory.)

Newman [32] gives a careful and involved definition of emergence, and demonstrates that being in the basin of attraction of a strange attractor in a chaotic dynamical system is an emergent property under this definition. His aim is simply to show the existence of *something* that is emergent under his definition; he is careful to emphasise that this is not necessarily the only kind of emergent property.

Goldstein [27] (who includes an interesting survey of the history of the concept of emergence) also links emergence and attractors: "emergence is associated with the arising of new attractors in dynamical systems (i.e., bifurcation)". It is these new attractors that "allow for the emergence of something radically novel".

Kaufmann [31] considers random boolean networks (RBNs), taken to be highly simplified models of gene regulatory networks (GRNs). He analyses the structure and stability of their attractor spaces, and draws an analogy between these attractors and cell types: maybe somehow cells 'are' the attractors of GRNs.

In this dynamical systems view, everything is process (motion on an attractor), but when it is viewed on a suitable timescale, it behaves like a particle. Abraham ([3], as cited in [28]) says "An attractor functions as a symbol when it is observed . . . by a *slow observer*." The observer does not see the intricate dynamics on the attractor, just the "averaged attributes", and so this dynamics becomes a "symbol", or atomic component, in its own right. What is lost is the internal structure; what remains is a stable pattern that becomes an entity in its own right.

And if everything is process, then things can be independent of the details of the underlying substrate (composition of the underlying levels): the same processes could be supported by different substrates. This gives us some hope of a *science* of emergence that is (relatively) independent of the details of specific implementations, and depends just on their dynamical properties (attractor structures).

3. Recognising and measuring emergence

3.1. Characteristics

Holland [29, p.3] states: "It is unlikely that a topic as complicated as emergence will submit meekly to a concise definition, and I have no such definition to offer." He spends the rest of the book discussing the phenomenon, in order that a definition can, well, 'emerge'.

We do not have the luxury of a few hundred pages to explore the concepts. So, rather than attempt a concise definition, we give some characteristics that emergent systems exhibit, and by which we might recognise them.

Far-from-equilibrium: The system is an open far-from-equilibrium dissipative process, that is, with a constant flow of matter, energy, or information (entropy) through it. Thus it exists in a context, or environment, which provides the material from which it organises its relatively stable pattern of existence. The properties of the system depend not purely on its own organisation, but also on this context, and the boundary (or initial) conditions.

Levels: The system has different levels, exhibiting different length- and timescales. The dynamics of the lower levels exhibits attractors; these attractors are identified with higher level emergent properties: extended low level processes become high level atomic states, with their own dynamics.

Languages: The system has a low level language L, used to describe the implementation, cast in terms of the local components and their local interactions, and a high level language H, used to describe the resulting system. H employs concepts distinct from those of L, and in particular, is cast in terms of more global concepts (concepts encompassing larger spatial or temporal scales). These distinct concepts are emergent properties. (*cf* Ronald *et al* [39].) The requirement that H be distinct from L eliminates Aristotle's 'mere heap', or simple aggregate, properties. The concepts of H are related to the attractor, and other, structures of the space defined by L and the initial conditions.

Rather than drawing some sharp and possibly arbitrary boundary between weak and strong emergence, we can use these characteristics to begin to *quantify* the strength of the emergence, based on the degree of distinction between languages L and H. Possibly quantifications include the use of compressibility metrics, or relative entropy.

3.2. Examples

Consider an L-system description of plant morphology [36]. The low level is that of parallel application of grammar production rules, with strings being rewritten. The high level system is a picture of a fractally-structured plant. The morphology is an emergent property of the rewrite rules, under a particular graphical rendering of the resultant string (corresponding to the integration environment, see section 5.3).

Consider Reynolds' flocking boids [38]. The low level is that of individual boid particles, moving under simple rules. The high level comprises the flock. The flocking behaviour is an emergent property of the rules governing the boids. Such flocks exist in a highly dynamic environment: all the other boids form the environment of each individual boid.

Consider Conway's GoL [26, 12]. The lowest level is that of the CA rules, with stationary cells switching on and off. Higher levels consist of concepts such as gliders, patterns of 'moving cells'. The gliders and other patterns are emergent properties of the GoL rules. These patterns are attractors (or attractors under translation) in local subspaces. Given the rules, whether or not gliders appear depends entirely on the initial conditions. The low level cells that constitute a glider change over time, comprising a 'flow of material' (cells) through the glider.

Consider an atomic transaction, implemented by multipart protocol messages, for example, the Mondex electronic purse protocol [41, 42]. The low level implementation is a collection of electronic purses, communicating via a multipart message protocol, individually incrementing and decrementing their balances in response to particular messages. The high level system is a group of purses engaging in atomic value transfer events. The atomic transfers are a (weakly) emergent property of the protocol message rules. The continual flow of transfer requests from the environment drives the activity of the system.

Consider a computer's RAM. The low level is of logic gates connected in feedback loops. The high level is a component with memory. The memory is an emergent property of the gates and their connections.

Consider computer programs in general. The lowest level is the electron flow. Higher levels correspond to higher level programming languages and concepts. The behaviour is driven by programs provided by the environment, which result in exquisite organisation of the electron flows. The program execution is an emergent property of these electron flows. Although an executing program can be considered as following a trajectory through its state space, programs are rarely analysed in terms of attractors. This is (partly) because their basins of attraction are very small: their trajectories are very fragile to small perturbations.

4. Engineering emergence

4.1. Is it possible?

This philosophical discussion gives us the hope, at least, that engineering emergence is not an impossible dream. Rather than, as some authors do, throwing up our hands in despair, and claiming that emergence is unpredictable and hence undesignable, we now have several concepts that can help us in our quest. In particular, we have exorcised the demon of surprise. The concepts of hierarchical levels, length- and timescales, attractors, languages, etc, give us a conceptual toolbox with which to progress.

The functional task is to go from some high level global system specification (in language H), to a low level design or implementation (in the conceptually distinct and local language L). There are also non-functional concerns, such as safety, robustness, and dependability.

Abbott [2] is one of the few authors who explicitly talk about emergent properties in engineering design terms: the high level emergent is the *abstract design*, and the lower level is the *implementation* of that design. Furthermore, "these abstract designs are neither derivable from nor logical consequences of their implementations": a creative design step is necessary.

Abbott [2] also examines the requirements for enabling multiple levels of emergence, a concern if we are to build non-trivial systems. He concludes that, for "highly leveraged design" of complex systems, "every time we build a new system, it should be built so that it becomes part of our environment", and that we should "design [all new systems] as infrastructure services and not just as bits of functionality." These new systems thereby form part of an enriched environment, and hence become available for further complex emergent systems development. This approach has interesting consequences for safety and dependability, since we will not only be building new systems exploiting such pre-existing and probably ill-understood infrastructure, but also be using that infrastructure in ways unanticipated by its designers.

4.2. Beyond refinement

The classical computer science approach to rigorous software development, more honoured in the breach than the observance, is *refinement*: a rigorous mathematical procedure for moving from abstract specifications to correct concrete designs. For example, the Mondex electronic purse protocol [41, 42] was proved correct in this manner. The procedure establishes a strong connection between the abstract and concrete descriptions, linking them with a retrieve relation. However, as we have noted above, in a (strongly) emergent system, the high level abstract language

H and the low level concrete language L employ distinct concepts. If these concepts are sufficiently distinct (for example, in the GoL H expresses properties of gliders moving and colliding, L expresses properties of stationary sites changing state) classical refinement techniques will not be able to establish the required connection [34].

Retrenchment [9] is a technique that weakens and generalises the classical refinement conditions, thereby allowing weaker relationships between high and low level systems to be established. It has been suggested as a technique applicable to emergent systems [8]. However, it is in some sense a first order perturbation of refinement, and may not be a powerful enough technique to cope with fully emergent systems: it still requires languages H and L to be somehow comparable.

Baas & Emmeche [7] suggest an approach based on category theory to link the high and low levels. However, arguments against the applicability of refinement apply also to this approach: the high and low level system descriptions simply do not exhibit this kind of crisp relationship.

Our own initial approach is based on the technique of *rule migration* [35, 43, 44, 45] to translate a high-level multi-layer design (with downward causation between the levels used to simplify the design description) into an 'equivalent' simple low level implementation with only local rules, exhibiting the design properties via upward causation as emergent behaviour.

5. TUNA

5.1. Nanotechnology

In the late 1950s, Richard Feynman delivered a visionary lecture at Caltech entitled "There's plenty of room at the bottom" [24]. He discussed the problem of manipulating and controlling things on an atomic scale. In 1986, Eric Drexler set out his view of the emerging field of nanotechnology [20]: nanites are nano-scale robots that collectively operate to cause macroscopic effects; nanotech assemblers are nanites that build artefacts.

Researchers have begun to discuss the technology required to build nanites [21], and distinct application areas have started to appear. Most work on nanotechnology is related to the physical construction of nanites; the systems engineering aspects have received significantly less attention than nanite fabrication.

Drexler's earliest work envisaged nanites under centralised control, commanded to build explicit artefacts. If instead nanite populations are viewed as complex systems of locally communicating agents, then the constructed artefacts are the *emergent properties* of the collective nanite operations. Software engineering challenges for the nanite designer are to design the local individual behaviours that produce the desired global emergent behaviour, whilst simultaneously ensuring that no *undesirable* global behaviour [25] emerges.

5.2. TUNA project overview

The TUNA ("Theory Underpinning Nanotech Assemblers") project is a feasibility study investigating engineering techniques to model, analyse, and simulate nanite systems, and that support the construction of safety cases for such critical mechanisms. We are investigating: (1) possible languages for developing abstract models of nanites and the protocols needed for inter-nanite communication and for external control; (2) methods for verifying and validating executable models of nanites; (3) the requirements for an appropriate simulation framework for executable models of nanites and of networks of nanites; (4) the nature of safety cases for the use of networks of nanites.

In carrying out these investigations, we are using a case study from the field of nanomedicine: the process of *haemostasis* (the staunching of bleeding) caused by mechanical platelets. We are using this as a motivating example, to explore modelling and simulation requirements, rather than producing detailed models of actual nanites. To this end, we have produced simple CA-based models, and mobile process models, of platelets forming clots. Two other papers in these proceedings describe other features of the TUNA project: Schneider *et al* [40] describe the abstract CSP || B formal models; Welch *et al* [48] describe the occam-π simulation framework. Here we discuss the engineering architecture.

5.3. An architecture

Engineering is taken, here, to mean development according to requirements that include not only functional aspects of the required system, but also dependability characteristics. Our work in TUNA on the theoretical engineering of nanite assemblers draws heavily on critical systems work on dependability and on macro-scale system-of-systems (SoS) assurance (for example, [4, 19, 46]).

Our goal is a framework for *assurance-driven engineering* of nanotech SoS, akin to Weaver *et al*'s [46] "evidence-based software engineering". In essence, for nanotech assembly to become commercially acceptable, there must be confidence in the dependability of the engineered nanites. There must be clear, defendable arguments that the nanites operate correctly in their intended environment, fail without causing harm, and cannot cause unintended emergent effects if operating outside their environment.

Initial work is investigating the adaptation of high-level dependability argument patterns [19] to the TUNA case study. In considering this, it is necessary to understand the

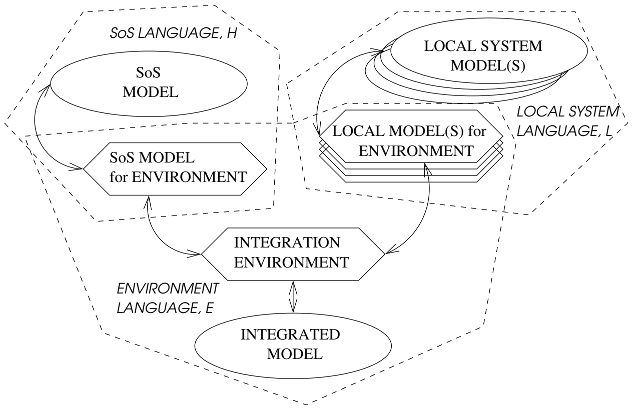

Figure 1. The engineering architecture for emergent SoS

generic requirements of nanoscale SoS. From characteristics of military SoS [4], aspects in common with nanotech SoS include: multiple, heterogeneous, mobile components, each of which is a system in its own right; *ad hoc* communication; potential for autonomy. Aspects of military SoS such as intent and local goals need further consideration.

An important contribution to the engineering of emergence is the identification of architectural elements of the system. We propose a three-element architecture (figure 1), comprising: the high-level system, or SoS, model; the local component (here nanite) model; the integrated model of the implementation, combining the local elements such that the desired system emerges [34]. These models are linked by an integration environment, a frame of reference that is accessible to both the L and H languages. Once the environment is known (or designed), the local and high-level models are mapped to suitable models expressed in the language of this environment (the hexagonal boxes in figure 1).

To illustrate the use of this architecture, we show how it is populated for four examples: (i) GoL gliders, (ii) TUNA CA platelet simulation, (iii) TUNA mobile process platelet simulation, (iv) nanite platelet system.

GoL gliders [34, 35]. The system is a collection of gliders moving and interacting. Each local component is a cell with a state; the state changes according to the GoL CA rules, which refer to the states of neighbourhood cells. The integration environment is a regular rectangular grid, defining the cells' positions. The system mapped to the environment comprises gliders moving relative to the grid locations. The components in the environment are sited on the grid and have their neighbourhood defined by adjacency on the grid. The integrated model is a grid of coloured, changing sites that exhibits glider patterns. (In [34], we additionally consider initialisation and detection of emergence. These are open questions, requiring further research.)

TUNA CA platelet simulation [35, 40, 47, 48]. The system is a collection of platelets moving and clotting. Each local component is a site with a state (states modelling the presence or absence of a platelet), changing according to the CSP CA-like rules, given inputs signalled along communication channels. The environment is a regular array of sites linked by communication channels, defining the sites' positions and communication links. The system mapped to the environment comprises platelets and clots moving relative to the array locations. The components in the environment are sited on the array and have their neighbourhood and communications defined by the array structure. The integrated model is an array of coloured, changing sites that exhibits moving platelet and clotting patterns.

TUNA mobile process platelet simulation [35, 44]. (Modelling movement with CA rules is complicated. In TUNA, we also use a higher-level modelling technique based on *mobile processes*.) The system is a collection of platelets moving and clotting. Each local component is a process modelling a platelet, changing according to local rules given inputs signalled along communication channels. The environment is a regular array of sites linked by communication channels, modelling the processes' physical locations. The system mapped to the environment comprises platelets and clots moving relative to the array locations. The components in the environment are linked to the array and have their neighbourhood and communications defined by the array structure. The integrated model is an array of mobile platelet processes that exhibit clotting patterns. (In [43, 45], we exploit the mapping for rule migration: moving from a mobile process model to a CA model by changing the local and environmental models whilst keeping fixed the system model.)

Nanite platelet system. The system is a collection of platelets moving and clotting. Each local component is a nanite platelet, changing according to local rules given inputs signalled through its sensors. The environment is the physical environment of the blood vessel, providing the nanites' physical locations, movement, and inputs. The system mapped to the environment comprises platelets and clots in the blood vessel. The components in the environment are situated in the blood vessel, and provided with inputs from the physical environment via their sensors. The integrated model is an array of mobile platelets that exhibit clotting behaviour.

The integration environment in the mobile process model simulates the physical environment of real nanites, providing location and input information. The nanite system exploits the given physical environment, which does not need to be implemented.

5.4. Simulations

A final observation on the engineering of emergence is that engineering a computer simulation is a very different SoS goal from that of real nanite assembly. The accuracy (relative to reality) of a computer simulation depends on the ability to program the computer to simulate the environment. Both the simulated environment and the simulated nanite can be modelled, verified and refined using conventional engineering approaches.

In assurance-driven engineering of a real nanite assembler, it would be necessary to have deep understanding of the aspects of the natural environment that interact with the nanites. This understanding could be assisted by simulations, but assurance from verification of the simulations does not map to assurance of the real SoS. Furthermore, the physics and chemistry of both the nanites and the environment become real actors in the assurance process, whereas these are usually ignored (or abstracted out) in computer simulation. In reality, the environment is a multiple-abstraction-level SoS with its own inherent emergent properties. (See also the discussion in [1], where Abbott argues that SoSs are 'open at the bottom', and that changes at the lowest implementation levels are often influenced by the environment.) Understanding how much of the environmental SoS can interact with the nanites may be the key to assurance-driven nanite engineering. This understanding must encompass the physical and chemical effects on the real nanites, with their potential to tamper with intended environmental interactions, analogous to side channel attacks via the hardware of the nanites.

6. Conclusions

Emergent properties will be important features of future engineered systems. Despite some authors' despair at the very idea, we believe it possible to engineer emergent systems in a principled manner. Here we have outlined our first steps towards that goal, demonstrating how our three-element architecture can be applied to emergent complex systems, but much yet remains to be done.

Acknowledgements

H. R. Turner is funded by an EPSRC/BAE Systems CASE Studentship. The TUNA project is supported by EPSRC grant EP/C516966/1. We thank the other members of the TUNA consortium (Ana Cavalcanti, Fred Barnes, Steve Schneider, Helen Treharne, Peter Welch, Jim Woodcock) for many lively discussions and sharing of ideas. We also thank Russ Abbot and Jon Timmis for helpful comments on an earlier draft.

References

[1] R. Abbott. Open at the top; open at the bottom; and continually (but slowly) evolving. In *IEEE International Conference on System of Systems Engineering, Los Angeles, USA*, Apr. 2005.

[2] R. Abbott. Emergence explained: getting epiphenomena to do real work. Feb. 2006. arXiv:cs/0602045.

[3] R. H. Abraham. Dynamics and self-organization. In F. E. Yates, A. Garfinkel, D. Walter, and G. Yates, editors, *Self-organizing Systems: The Emergence of Order*, pages 599–613. Plenum, 1987.

[4] R. Alexander, M. Hall-May, and T. Kelly. Characterisation of systems of systems failures. In *22nd Int. Systems Safety Conference*, Aug. 2004.

[5] P. W. Anderson. More is different. *Science*, 177(4047):393–396, 1972.

[6] Aristotle. *Metaphysics*, volume book H (VIII). 350 BC. Translation from W. D. Ross, *Aristotle's metaphysics*, 2 vols, Oxford University Press, 1924.

[7] N. A. Baas and C. Emmeche. On emergence and explanation. *Intellectica*, 25(2):67–83, 1997. Also published as SFI Working Paper 97-02-008.

[8] R. Banach, C. Jeske, S. Fraser, R. Cross, M. Poppleton, S. Stepney, and S. King. Approaching the formal design and development of complex systems: The retrenchment position. In *Workshop on Software and Complex Systems, 9th IEEE International Conference on Engineering of Complex Computer Systems, Florence, Italy*, 2004.

[9] R. Banach and M. Poppleton. Retrenchment: An engineering variation on refinement. In D. Bert, editor, *2nd International B Conference, Montpellier, France, April 1998*, volume 1393 of *LNCS*, pages 129–147. Springer, 1998.

[10] M. A. Bedau. Weak emergence. In J. Tomberlin, editor, *Philosophical Perspectives 11: Mind, Causation, and World*, pages 375–399. Blackwell, 1997.

[11] R. D. Beer. A dynamical systems perspective on agent-environment interaction. *Artificial Intelligence*, 72:173–215, 1995.

[12] E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning Ways for Your Mathematical Plays Volume 2: games in particular*. Academic Press, 1982.

[13] M. H. Bickhard and D. T. Campbell. Emergence. In P. B. Andersen, C. Emmeche, N. O. Finnemann, and P. V. Christiansen, editors, *Downward Causation*, pages 322–348. Aarhus University Press, 2000.

[14] A. Burns, I. J. Hayes, G. Baxter, and C. J. Fidge. Modelling temporal behaviour in complex socio-technical systems. Technical Report YCS-2005-390, Department of Computer Science, University of York, 2005.

[15] D. T. Campbell. 'Downward Causation' in hierarchically organised biological systems. In F. J. Ayala and T. Dobzhansky, editors, *Studies in the Philosophy of Biology: reduction and related problems*, chapter 11, pages 179–186. Macmillan, 1974.

[16] R. J. Campbell and M. H. Bickhard. Physicalism, emergence and downward causation. 2001. www.lehigh.edu/˜mhb0/physicalemergence.pdf.

[17] A. Clark. *Being There: putting brain, body and world together again*. Oxford University Press, 1997.

[18] J. P. Crutchfield. The calculi of emergence: Computation, dynamics, and induction. In *Physica D special issue on the Proceedings of the Oji International Seminar*, 1994.

[19] G. Despotou and T. Kelly. Extending the safety case concept to address dependability. In *22nd Int. Systems Safety Conference*, Aug. 2004.

[20] K. E. Drexler. *Engines of Creation*. Anchor Books, 1986.

[21] K. E. Drexler. *Nanosystems: Molecular Machinery, Manufacturing, and Computation*. Wiley, 1992.

[22] C. Emmeche, S. Køppe, and F. Stjernfelt. Explaining emergence: towards an ontology of levels. *Journal for General Philosophy of Science*, 28:83–119, 1997.

[23] J. Faith. Why gliders don't exist: Anti-reductionism and emergence. In C. Adami, R. Belew, H. Kitano, and C. Taylor, editors, *Artificial Life VI: Proceedings of the Sixth International Conference on Artificial Life*, pages 389–392, 1998.

[24] R. P. Feynman. There's plenty of room at the bottom. In A. Hey, editor, *Feynman and Computation: exploring the limits of computers*, pages 63–76. Perseus Books, 1999.

[25] R. A. Freitas Jr. Some limits to global ecophagy by biovorous nanoreplicators, with public policy recommendations. Foresight Institute, Apr. 2000.

[26] M. Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "Life". *Scientific American*, 223(4):120–123, October 1970.

[27] J. Goldstein. Emergence as a construct: History and issues. *Emergence*, 1(1):49–72, 1999.

[28] J. Goldstein. Emergence: A construct amid a thicket of conceptual snares. *Emergence*, 2(1):5–22, 2000.

[29] J. H. Holland. *Emergence: from chaos to order*. Oxford University Press, 1998.

[30] W. Hordijk, J. P. Crutchfield, and M. Mitchell. Mechanisms of emergent computation in cellular automata. In A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature V*, pages 613–622. Springer, 1998.

[31] S. A. Kauffman. *The Origins of Order: self-organization and selection in evolution*. Oxford University Press, 1993.

[32] D. V. Newman. Emergence and strange attractors. *Philosophy of Science*, 63(2):245–261, 1996.

[33] T. O'Conner. Emergent properties. *Americal Philosophical Quarterly*, 31:91–104, 1994.

[34] F. Polack and S. Stepney. Emergent properties do not refine. In *REFINE 2005 Workshop, Guildford, UK, April 2005*, volume 137 of *ENTCS*, pages 163–181. Elsevier, 2005.

[35] F. Polack, S. Stepney, H. Turner, P. Welch, and F. Barnes. An architecture for modelling emergence in CA-like systems. In M. S. Capcarrere, A. A. Freitas, P. J. Bentley, C. G. Johnson, and J. Timmis, editors, *Advances in Artificial Life: ECAL 2005, Canterbury, UK, September 2005*, volume 3630 of *LNAI*, pages 433–442. Springer, 2005.

[36] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer, 1990.

[37] P. Rendell. Turing Universality of the Game of Life. In A. Adamatzky, editor, *Collision-Based Computing*. Springer, 2002.

[38] C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics (SIGGRAPH '87 Conference Proceedings)*, 21(4):25–34, 1987.

[39] E. M. A. Ronald, M. Sipper, and M. S. Capcarrère. Testing for emergence in artificial life. In D. Floreano, N. J.-D., and F. Mondada, editors, *Advances in Artificial Life: 5th European Conference*, volume 1674 of *LNCS*, pages 13–20. Springer, 1999.

[40] S. Schneider, A. Cavalcanti, H. Treharne, and J. Woodcock. Bloody CSP: a layered behavioural model of platelets. In *ICECCS 2006*. IEEE, 2006. (these proceedings).

[41] S. Stepney, D. Cooper, and J. Woodcock. More powerful Z data refinement: pushing the state of the art in industrial refinement. In J. P. Bowen, A. Fett, and M. G. Hinchey, editors, *The Z Formal Specification Notation, 11th International Conference of Z Users, Berlin, Germany, September 1998*, volume 1493 of *LNCS*, pages 284–307. Springer, 1998.

[42] S. Stepney, D. Cooper, and J. Woodcock. An electronic purse: Specification, refinement, and proof. Technical monograph PRG-126, Oxford University Computing Laboratory, July 2000.

[43] H. Turner and S. Stepney. Rule migration: Exploring a design framework for modelling emergence in CA-like systems. In *ECAL Workshop on Unconventional Computing*, 2005.

[44] H. Turner, S. Stepney, and F. Polack. A simulation environment for emergent properties (extended abstract). In *ECCS 2005: European Conference on Complex Systems, Paris, France*, 2005.

[45] H. Turner, S. Stepney, and F. Polack. Rule migration: A design framework for emergence. *International Journal of Unconventional Computing*, 2(4), 2006.

[46] R. Weaver, G. Despotou, T. Kelly, and J. McDermid. Combining software evidence — arguments and assurance. In *REBSE05, St Louis, USA*. ACM, 2005.

[47] P. Welch and F. Barnes. Mobile Barriers for occam-pi: Semantics, Implementation and Application. In J. Broenink, H. Roebbers, J. Sunter, P. Welch, and D. Wood, editors, *Communicating Process Architectures 2005*, volume 63 of *Concurrent Systems Engineering Series*, pages 289–316. IOS Press, Sept. 2005.

[48] P. H. Welch, F. R. M. Barnes, and F. A. C. Polack. Communicating complex systems. In *ICECCS 2006*. IEEE, 2006. (these proceedings).