From Computational Thinking to Systems Thinking:

A conceptual toolkit for sustainability computing

Steve Easterbrook
Dept of Computer Science
University of Toronto
140 St George Street, Toronto, Ontario, Canada
Email: sme@cs.toronto.edu

***Abstract*—If information and communication technologies (ICT) are to bring about a transformational change to a sustainable society, then we need to transform our thinking. Computer professionals already have a conceptual toolkit for problem solving, sometimes known as *computational thinking*. However, computational thinking tends to see the world in terms a series of problems (or problem types) that have computational solutions (or solution types). Sustainability, on the other hand, demands a more systemic approach, to avoid technological solutionism, and to acknowledge that technology, human behaviour and environmental impacts are tightly inter-related. In this paper, I argue that *systems thinking* provides the necessary bridge from computational thinking to sustainability practice, as it provides a domain ontology for reasoning about sustainability, a conceptual basis for reasoning about transformational change, and a set of methods for critical thinking about the social and environmental impacts of technology. I end the paper with a set of suggestions for how to build these ideas into the undergraduate curriculum for computer and information sciences.**

***Index Terms*—Computers and Society, Computer Science Education, Computational Sustainability.**

I. Introduction

The remarkable growth of information and communications technology (ICT) over the past two decades poses many serious dilemmas for sustainability practitioners. On the one hand, ICT has grown to become a significant fraction of humanity's environmental footprint, through demand for energy, particularly the embodied energy of manufacture [1], demand for scarce or critical metals and minerals [2], and the growing problem of disposal of e-waste [3]. On the other hand, ICTs are frequently cited as a key part of the solution, as they offer new opportunities to monitor and analyze human activity to guide us to low impact choices, and to replace resource-intensive activities (e.g. movement of people and goods) with alternatives (e.g. teleconferencing and virtualization) [4].

Unfortunately, not only is western society a long way from sustainability, analysis shows that we cannot achieve sustainability merely by improving energy efficiency – we will need a dramatic reduction in resource consumption [5]. Paradoxically, ICTs are part of the drive to ever growing consumption, as the combined effects of Moore's law and built-in obsolescence shorten produce lifetimes for hardware, while the desire for greater connectivity accelerates demand for new gadgets.

This growing consumption of ICT products is driven, in part, by an alarming set of technology industry trends, all of which push society further away from a sustainable level of consumption of energy and material goods. These are largely unacknowledged in the mainstream computing literature:

- A computer industry that sells gadgets with ever shorter shelf-lives, without regard to environmental and social impact of their manufacture, and disposal of the resulting e-waste [6].
- A tendency towards *technological solutionism*, which treats complex societal problems in a simplistic way, such that the solution can be the sale of a new app, new web service, or a new device, without exploring the broader environmental consequences [7].
- A tendency towards automating and optimizing existing solutions without first exploring their social and environmental impacts, thus reducing resilience and locking us further into unsustainable ICT infrastructures [8].
- A preference for moving ever more of the internet technology stack into proprietary software ecosystems, which prevents users from adapting or re-designing their technology to suit local needs and local contexts [9], [10].

Sustainability is an emergent phenomenon from the interaction between (very large numbers of) people, and the ways in which we build and use technology. Innovations in ICT to improve sustainability may be counter-productive in the long term, if they contribute to these underlying trends in consumption. Hence, sustainability requires a whole systems approach [11]. I argue that such an approach must address:

- The interconnections between the grand challenges of climate change, food production, water management, pollution, environmental health, and the end of cheap energy. None of these problems can be tackled independently.
- An understanding that these represent a set of *dilemmas* to which we can respond, rather than problems that we can solve, and that any attempt to solve them will give rise to further, unanticipated problems.
- The idea that unchecked growth in any one technology is unsustainable on a finite planet, and that therefore any technology that becomes too successful must eventually threaten sustainability, due to overuse and co-dependence.
- An acknowledgement that the human activity systems in

which technology sits create their own dynamics, and it is the dynamics of these social systems that shape our dependence on technology, and limit the independent behaviour of human actors. As software infrastructure channels ever more of our social behaviour, we inadvertently create feedback loops that prevent transformational change and frustrate attempts by individuals to adopt more sustainable lifestyles

In this paper I argue that the failure to think systemically is a critical weakness in our understanding of the transformations needed to achieve sustainability. Many of the ways that we seek to apply ICTs as part of a solution risk falling into this trap. I argue that such a trap arises because of how we educate ICT professionals. I identify *computational thinking* as a major limiting factor, and propose to supplement it with *systems thinking*, integrated throughout the ICT curriculum. I end the paper with some examples of how to implement this suggestion.

II. Computational Thinking

A. What is Computational Thinking?

Computer scientists tend to approach problems in a particular way. Because programming is fundamental to computer science education, computer scientists tend to think like programmers. That is, they look for algorithmic solutions to problems, in terms of data manipulation and process control. In a widely cited paper in 2006, Jeanette Wing termed this *computational thinking* [12], and argued that this practice may be the most important contribution computer science makes to the world, and that it should be taught to all students in all disciplines.

Wing's original paper did not offer a succinct definition for computational thinking, but offered many examples of how computer scientists tackle common problems: *"When your daughter goes to school in the morning, she puts in her backpack the things she needs for the day; that's prefetching and caching. When your son loses his mittens, you suggest he retrace his steps; that's back-tracking. [...] Which line do you stand in at the supermarket?; that's performance modeling for multi-server systems."* [12]. Extrapolating from these examples, the overall message is that computer scientists have a toolbox of methods for matching problem situations to standard types of solution, drawn from various parts of the computer science curriculum, and, perhaps just as importantly, a standard terminology to describe these abstract problem-solution patterns.

Encouraged by funding programs from the NSF, the US computer science community has readily adopted the term computational thinking, using it as a slogan to re-design existing computer science curricula to make them more attractive to students, and to develop new courses aimed at audiences who would not otherwise be exposed to computer science. For example, the Computer Science Teachers Association (CSTA) set up a task force to *"explore and disseminate teaching and learning resources related to computational thinking"*. This task force offers the following definition [13]:

"Computational thinking (CT) is a problem-solving process that includes (but is not limited to) the following characteristics:

- *Formulating problems in a way that enables us to use a computer and other tools to help solve them.*
- *Logically organizing and analyzing data.*
- *Representing data through abstractions such as models and simulations.*
- *Automating solutions through algorithmic thinking (a series of ordered steps).*
- *Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources.*
- *Generalizing and transferring this problem solving process to a wide variety of problems."*

The task force goes on to add that these skills are supported by a set of broader attitudes, including ability to deal with complexity and open ended problems, tolerance for ambiguity, and ability to work with others to achieve a common goal. While other definitions have been proposed, all seem to agree that computational thinking focusses on the application of a set of problem abstractions: *"Today the term has been expanded to include thinking with many levels of abstractions, use of mathematics to develop algorithms, and examining how well a solution scales across different sizes of problems"* [14].

B. What's Wrong with Computational Thinking?

Since the concept was introduced, there has been remarkably little critical thinking about computational thinking. The few critiques that have been written tend to focus on either the vagueness of the term [15], or on a concern that the field of computer science should not be reduced to just one of its practical tools: *"Computational Thinking is one of the key practices of computer science. But it is not unique to computing, and is not adequate to portray the whole of the field"* [14].

However, a deeper critique is called for. If computational thinking is the central tool of computer scientists, then we ought to consider whether computational thinking becomes just another instance of Maslow's Hammer [16]: *"If all you have is a hammer, then everything looks like a nail"*. In other words, computer professionals may attempt to solve all problems through algorithmic means, while failing to perceive those that cannot be expressed using the abstractions of CT.

The computational thinker looks for problems that can be tackled with computers. Immediately, this provides a selective lens through which to view the world. Problems that are unlikely to have computational solutions (e.g. ethical dilemmas, value judgements, societal change, etc) are ignored. Others are reduced to a simpler, computational proxy. It is no coincidence that computer science students tend to be less morally mature than students from other disciplines [17]. Ethical dilemmas have no computational solutions, and so are overlooked when peering through a CT lens.

At heart, CT is inherently reductionist. Computational problems are tackled by reducing them to a set of discrete variables that can be mapped onto abstract data types, and a set of algorithmic steps for manipulating these data types. In the process, multiple perspectives on the nature of the problem are lost, as is any local, contingent knowledge about the problem situation [18]. Computational thinking thus ignores the fact that any particular expression of the "the problem to be solved" is the result of an ongoing negotiation between the competing needs of a variety of stakeholders [19], [20].

Descriptions of computational problems are accepted uncritically, as long as they can be *"formulated in a way that enables us to use a computer ... to solve them"*. This means that students of CT lack awareness of the political context in which these "problems" arise. For example, a series of studies widely reported in the media, and dating back at least a decade, have vastly over-stated the energy footprint of smartphones and computing equipment [21]. These studies have been accepted uncritically across the industry, because they serve a number of interests. The coal industry sponsored these studies to persuade people that high demand will mean phasing out coal-fired power stations is an unrealistic policy goal in the near future. The computing industry uses them to persuade consumers to replace their existing devices with new "greener" gadgets. The media uses them to create a narrative of shocking new findings, to boost readership. Meanwhile, the focus on computing devices as "energy hogs" diverts attention away from more serious impacts, such as mining for rare and critical minerals, and disposal of e-waste. The computational thinker lacks the conceptual toolkit to understand these issues.

Reductionism tends to eshew complexity. In the CSTA definition above, "dealing with complexity" is added as a supporting attribute to computational thinking. However, in computer science curricula, the only kind of complexity is *algorithmic complexity*, which offers little insight into the broader study of complex adaptive systems [22]. In his review of complexity science, Manson suggests that *"algorithmic complexity offers two relatively ancillary contributions to complexity theory overall"* [23], namely a measure of the effort needed to solve a mathematical problem, and an understanding of the limits to storing and communicating data, provided by information theory. He identifies two other forms of complexity needed for understanding the behaviour of complex, non-linear systems: the *deterministic complexity* of chaos theory and catastrophe theory; and the *aggregate complexity* that arises with emergent behaviour of the interaction of many components within a system. Both of these forms of complexity can be usefully studied through computational means, primarily through the use of simulation models, but CT, at least as defined in the literature, does not provide a suitable set of concepts by which to do this.

Reduction of problems to their computational components often leads to practices that undermine sustainability. Consider for example, the importance attached to process optimization in CT, which arises because of the way in which computational thinkers deal with scaling. If you solve a problem by reducing it to a set of basic abstractions, you then have to scale up your computational solution in the face of two fundamentally limited resources: memory space and processor time. Hence, computer scientists are trained to identify computational solutions that optimize for space and time bounds. This seduces computational thinkers to treat optimization as a universally good idea, and apply it in places where it is inappropriate.

Traffic management in a modern city offers an illustrative example. Computational thinkers tend to approach traffic congestion as an optimization problem: by collecting more data about traffic volumes, and using it to dynamically adjust the traffic signals at each intersection, the idea is that congestion can be reduced by optimizing traffic flow. Unfortunately, this solution usually has the opposite effect: congestion increases. This is because congestion is a non-linear property of the relationships between traffic volumes, vehicle speed, and road capacity. Spare capacity is crucial, because it acts as a buffer during times of high traffic volume. If you optimize the traffic flow, you remove this spare capacity, and hence reduce the overall resilience of the system to small variances in demand, and a perceptible improvement in throughput tends to trigger a an increase in demand anyway. More generally, systems tend to acquire resilience through redundancy – having multiple pathways by which to achieve a result [24]. They lose resilience through resource optimization – a system that is operating at maximum throughput has no capacity for dealing with minor shocks.

C. The Rise of Technological Solutionism

The above critique explains the prevalence of *technological solutionism*, which is widespread in the software industry today. Morozov takes aim at this in his recent book *"To Save Everything, Click Here"* [7]. The title of the book is a play on the belief that Silicon Valley will help us tackle some of the world's biggest challenges (hunger, poverty, ecosystem destruction, climate change) by offering us new apps for our smartphones. Morozov defines solutionism as the belief that complex societal problems can be re-cast as *"neatly defined problems with definite, computable solutions or as transparent and self-evident processes that can be easily optimized—if only the right algorithms are in place!"*.

However, Morozov's point is more subtle than merely observing that computational thinking has its limits. His key point is that our relationship with technology re-shapes our "problem-solving infrastructure" in unexpected ways. For example:

- The idea of *gamification* (changing social behaviours by offering rewards and incentives, to turn it into a game) is dangerous because it trains us to think in terms of regulating the individual citizen, rather than regulating the broader systems in which we live: the industries and governments that hold real power.
- The idea of *big data* (the belief that we can optimize societal processes by collecting and analyzing ever larger sets of data on human behaviour) is dangerous because it trains us to believe that the solutions to our problems can be found through automated pattern matching over ever more intrusive measurement of our behaviour, rather than

in improving our understanding of the forces that shape that behaviour.

- The idea of *crowdsourcing* is dangerous because it tends to undermine our belief in the value of expertise. The successes of crowdsourcing, such as the more popular sections wikipedia, tend to overshadow its woeful inadequacy in areas where deep expertise is needed, such as medical advice [25].

Success of these approaches on smaller, limited problems in turn tends to increase our confidence that technological solutionism works, thus reducing our ability to address more fundamental challenges.

The term *solutionism* also serves to emphasize the point that thinking in terms of problems and solutions is itself often counter-productive. In their classic 1973 paper, *"Dilemmas in a General Theory of Planning",* Rittel and Webber [26] contend that many societal issues are better thought of as *dilemmas*, to which we should respond intelligently, rather than as *problems* that we can solve. They distinguish between '*tame*' problems, which have a clear, definitive problem formulation and the possibility of an objectively correct solution, and '*wicked*' problems (or "dilemmas"), which are deeply embedded in a complex problem situation. Wicked problems have the following properties:

- *No definitive formulation*. Often, the only way to fully understand the problem is to attempt to solve it, and so we cannot clearly state the problem in advance.
- *No stopping rule*. As there is no end to the causal chains that link open systems, the only limit to solving a wicked problem is a value judgement that "this is as much as we can reasonably achieve".
- *Potential solutions must be evaluated* for whether they are better or worse, rather than whether they are correct.
- *It is not possible to fully evaluate a solution* in any reasonable timeframe. After implementation, the waves of consequences may go on indefinitely, giving us no way of tracing all such consequences at any given timepoint.
- *Every solution is a one-shot operation*. Each attempted solution changes the nature of the problem, so there is no opportunity to attempt to learn by trial and error.
- *It is not possible to enumerate all potential solutions*.
- *Every wicked problem is essentially unique*. While a specific problem might be similar to previous problems, the local contextual factors vary, and these are significant enough to prevent generic solutions from working.
- *Every wicked problem can be treated as a symptom of another problem*.
- *There are multiple ways of explaining the nature of the problem*, and the choice of description will determine the nature of acceptable solutions. Furthermore, the choice of explanation is usually determined by the worldview of the person describing the problem.

Failure to understand the nature of wicked problems is widespread across many disciplines. For example, classical economic theory is often criticized for its dependence on abstract mathematical models that capture idealized behaviour rather than what actually happens in the world, and for adherence to global indicators, such as GDP, that are increasingly divorced from our actual sense of prosperity [27], [28].

However, I have focussed this critique on CT for two reasons. First, no such critique currently exists in the literature. Second, advocates of CT present it as a central paradigm for education of ICT professionals, at precisely the time when we need computing practitioners to have a broader awareness of the social and environmental consequences of their work. Over-reliance on computational thinking is likely to lead not just to computational solutions that ignore social and environmental sustainability, but often to solutions that actively undermine such sustainability.

III. Sustainability as a System Condition

Sustainability itself is a wicked problem that resists definitive formulation. Many definitions have been proposed. The central idea is the ability of a system to endure, perhaps indefinitely, but there is little consensus on what a sustainable human society would look like, beyond the observation that, however you measure it, our current approach to global energy and resource management is unsustainable [5].

This core idea traces back to early work in ecology, which identified the self-sustaining nature of ecosystems. For example, water and nutrient cycles form self-replenishing systems that can operate indefinitely. Food chains tend to be self-sustaining, with predator-prey relationships keeping the relative populations of each species in balance. Interactions between species form a complex web, so the relative numbers of each species can vary dramatically over time, but, as a whole, such ecosystems can be remarkably resilient.

Successful ecosystems endure because this web of interactions between species give rise to self-balancing *feedback loops*: too much change in one direction triggers an increasing pushback in the opposite direction. If one species becomes too numerous, its food supply dwindles, and its predators become more numerous, until the numbers are pushed back into balance. Indeed, the very fact that we can identify an ecosystem as a coherent whole that endures over time depends on the existence of these balancing feedback loops.

While balancing feedback loops allow an ecosystem to survive within some normal range of natural variability, this does not necessarily provide it with the *resilience* to survive shocks imposed by interaction with other systems [29]. For example, an invasive species transplanted from another region can disrupt an ecosystem so that the natural feedback processes are overwhelmed. Similarly, excessive pollution or a changing climate can expose an ecosystem to conditions from which it can no longer recover.

These principles apply more generally: any system that endures long enough to be observed and described must have self-balancing feedback loops constraining its behaviour. In economics, the demand for and price of goods are constrained by a feedback loop: if demand rises too high, so do prices, which eventually brings demand back down again. In a democratic society, when a government steers too far from mainstream opinion, voters react by voting for an alternative. In cultural terms, ideas that are too radical or shocking inspire a counter-movement aimed at resisting them, sometimes violently.

Moreover, different feedback loops operate at the same time, on different timescales. For example, on a longer timescale, growth in human emissions of carbon dioxide will eventually trigger a dramatic reduction in fossil fuel use, either voluntarily, through a deliberate global strategy of decarbonization, or catastrophically, through a collapse of industrial society as the disruption of severe climate change escalates [30]. However, on a shorter timescale, the push for political action on climate is met with increasingly shrill opposition, as people regard it as too expensive or too radical, and instead respond by re-enforcing cultural values that reject the science [31]. Thus, feedback loops might drive a system to exhibit a remarkably resilient behaviour (e.g. climate change denial) at one timescale, even as that behaviour undermines its resilience (e.g. climate change preparedness) at a longer timescale.

The more optimistic analyses of sustainability suggest that, at least in principle, it is possible to sustain a human population much larger than the current one, with a higher level of technological support, but only if we can construct the necessary technological infrastructure at a much faster rate than the growth in demand from a rising population [32]. For example, climate change would not be a problem today if several decades ago we had set out to build a renewable energy infrastructure at a rate that outpaced the growth in demand for energy, so that overall global emissions fell year-by-year, instead of rising. As we failed to do this, the challenge today is much greater, because we start from a much higher annual rate of emissions, and also have to replace the accumulated fossil fuel infrastructure we have been building in the meantime. The majority of the warming we can expect over the coming decades is determined by the rate at which we can replace this existing infrastructure [33]. In other words, our problem is not whether we can build a zero-carbon society, but whether we can build it *fast enough*.

These issues can be understood more readily when expressed in terms of *rates of flow*. For example, problems arise when the rate of production of waste overwhelms the rate that the environment can absorb them. While the carbon dioxide produced by burning fossil fuels is steadily sequestered in soils and ocean sediments, the rate of flow into these carbon sinks is dramatically slower than our current rate of production. Hence, an understanding of sustainability must incorporate an understanding of *rates of flow*, and how these in turn drive *rates of growth* [34].

In the long run, this analysis suggests a set of fundamental constraints on global society as a system, first formulated in this way by Daly [27]:

1) A sustainable society cannot use renewable resources faster than they can be replenished.
2) A sustainable society cannot generate wastes faster than they can be absorbed by the environment.
3) A sustainable society cannot rely on continued use of a non-renewable resource[1].

More recent work on sustainable development has suggested a fourth condition, that emphasizes the sustainability of social structures, for example by ensuring there is no systemic erosion of trust in society [35]. These conditions provide a starting point for exploring how to build a sustainable society, by framing sustainability as a condition of the system as a whole.

IV. Systems Thinking

In the previous section I presented sustainability in terms of system-level properties such as feedback loops and rates of flow. My purpose is not to suggest that this is the right way to define sustainability, but to point out that to reason about sustainability, we need to understand systems and their emergent properties. This forms the basis of the argument that *systems thinking* is an essential component of any attempt to bring about transformational change to a sustainable society. More specifically, I argue that systems thinking can overcome the weaknesses of computational thinking as a conceptual basis for designing ICT for sustainability.

More detailed introductions to the basics of systems thinking can be found elsewhere[2]. Here I concentrate on three specific gaps in computational thinking for which systems thinking provides an appropriate conceptual toolkit.

A. Domain Ontology for Sustainability Thinking

Computational thinking provides an ontology of computational concepts, and a set of terms for talking about them. For example, procedural and data abstractions provide the building blocks of computational solutions, and sequential and parallel composition provide a way of putting them together. Hierarchical decomposition is used to reduce complex problems, and encapsulation is used to create re-usable solutions. However, this ontology focusses on computational *solutions*, usually at the expense of detailed analysis of problems and problem contexts.

The subfield of Requirements Engineering (RE) extends the ontology of computational concepts for use in problem analysis, and hence comes closest to adopting a systems view [37]. In particular, RE adds concepts such as stakeholders, goals (and the satisfaction or denial of goals), scenarios, tasks, and means-ends analysis. It also provides tools for modelling problem situations, such as enterprise-level activities, and interdependencies between actors in an organization. However, the extent of the influence of computational thinking can be seen here as well, as the modelling notations of RE are dominated by computational metaphors, most notably, processes and their sequential and parallel composition.

[1]Note: this condition does not mean that we can never make use of non-renewable resources, such as fossil fuels. However, it means that using them is only ever a temporary arrangement, and hence their use should be carefully managed to provide an investment for their eventual replacement by renewable alternatives. For example, it might be reasonable to extract oil to provide the energy needed to kick-start a solar energy economy.

[2]e.g. Meadows [36] provides an excellent primer.

What is missing here is a set of concepts for reasoning about the dynamical behaviour of systems (both physical systems and socio-economic systems), irrespective of whether those systems include computational elements. Without this, computational thinkers tend to adopt an overly-simplistic, linear approach to cause-and-effect, for example by reasoning about before-and-after situations: a requirements model that expresses a problem to be solved and a design model that expresses a computational solution, with perhaps some acknowledgement that iteration over these will be necessary. The fact that many of the problems thus tackled are *wicked* is sometimes acknowledged but rarely addressed. Hence, computational solutions are rarely evaluated over multiple timescales or from multiple perspectives, and the systems behaviours that may undermine a promising solution are ignored.

Systems Thinking addresses this gap through a set of concepts for understanding and reasoning about system behaviour. These concepts include:

- **Stocks and Flows**. Often, the simplest way to begin a systems analysis is to explore it in terms of *stocks* (quantities that vary over time) and *flows* (inputs and outputs to stocks that affect their levels). Stocks may be concrete quantities, such as the level of water in a lake, or the migration of people to a city. Or they may be abstract quantities, such as beliefs or desires, social relationships, or wealth. As I noted above, many sustainability issues can be expressed as mis-matches between rates of flow, and hence this perspective offers an important tool for understanding and evaluating proposed sustainability projects.
- **Emergent behaviour**. This is the observation that systems tend to have properties that cannot be traced to individual components or groups of components, but rather arise from the interaction of those components with the entire system. For example, the *rebound effect* is an emergent property of the interaction of energy efficiency measures with human economic behaviour, so that money savings from the efficiency measures are often spent on other energy-intensive activities, and the full expected savings are rarely experienced in practice [38]. Social media offers many examples of emergent behaviour; for example it makes it harder to debunk misinformation, because of the way people collect and share memes that reinforce their existing worldviews [39]. People rarely use ICT in the way its designers expect, but computational thinking does not address this problem. While it is often hard to anticipate emergent behaviours, useful insights can be gained by abandoning reductionism, thinking holistically in terms of complex adaptive systems, and modeling their behaviour using non-linear dynamics [40].
- **Feedback loops**. *Reinforcing* feedback loops tend to amplify any change within the system through a chain of cause-and-effect that eventually delivers further change in the same direction. *Balancing* feedback loops tend to resist change, through a chain of cause and effect that eventually produces a change in the opposite direction. Feedback loops are a key concept in control theory, taught in engineering programs, but rarely in computer science. This is unfortunate, because they offer a powerful conceptual tool for understanding how the structure of a system constrains its behaviours. Coupled with a stock-and-flow analysis, feedback loops also offer a way of reasoning about exponential growth and its limits.

B. *Theories of Change*

A major failing of computational thinking is that it offers no conceptual toolkit for reasoning about how change happens in complex systems, other than perhaps an *information deficit* model — the idea that change occurs when the right information is provided to the right people at the right time. Hence, computational solutions to sustainability often focus on providing information in new forms, or in using new sources of information to provide new ways of controlling other technology. Computational thinkers also tend to assume that change happens through innovation (i.e. the creation of a new computational solution), accompanied by an assumption that individuals have agency over their social and environmental impacts, and that all they need are better tools to help them become more sustainable.

In contrast, systems thinking provides a number of rich theories of change. For example, Meadows uses the concepts of stocks and flows and feedback loops to construct a detailed analysis of *Leverage Points* [41]. This theory offers an explanation of why some attempts to intervene in a system are more likely to bring about change than others, and a typology of different kinds of systemic intervention. In general, changes that alter the underlying structure of the feedback loops are more likely to generate changes in systems-level behaviour than changes to the values of variables within the system. Furthermore, the highest leverage often comes from changes in how we think about the system, in terms of the goals we adopt and the mental models we use. The theory also explains why the most effective kind of leverage point are also often the hardest to use, because they attack the most deep-seated structure of a system.

Systems thinking also provides an explanation of *Path Dependence* and *Lock-In* [8]. The behaviour of a system is rarely random, but depends on the structure of its feedback loops. That structure is shaped by past experience, so that systems often settle into behaviours that depend not just on the current state, but on the path the system took to get there. Simple examples include the way some inventions (e.g. the QWERTY keyboard; the internal combustion engine) become dominant and then remain so, even once better solutions exist. Lock-in is particularly problematic when we seek transformational approaches to sustainability, because it is nearly always easier to build something that depends on the existing technological infrastructure (and hence entrenches it even further), than it is to attempt to transform that infrastructure. A canonical example is the difficulty of introducing electric cars, because of the lack of a refuelling infrastructure. For ICT, large software corporations often try to exploit lock-in, by developing technologies that do not inter-operate with other software ecosystems, thus locking customers into a proprietary software ecosystem [10]. These proprietary software ecosystems pose a major roadblock for sustainability, because they are explicitly designed to resist broader systemic change.

A third theory of systems change draws on ideas from *Chaos Theory*. Systems are often observed to have stable patterns of behaviours that endure over time, induced by their feedback loops. A system can bounce back after a catastrophic event, if the structures that created the system in the first place still endure. For example, a forest re-grows after a forest fire, because the conditions that favoured forest growth still exist. This allows a system to be *resilient* in the face of some types of change [24]. However, small changes can grow over time to overwhelm these feedback loops and drive the system to a new set of behaviours. To an outside observer, the system seems to flip suddenly from one regime to another. However, the *tipping point* is induced by an accumulation of small changes within the system that may be invisible to outside observers. As with emergent properties, tipping points may be easy to identify after the fact, but hard to predict [42].

Perhaps the most comprehensive theory of change in systems thinking is the *Panarchy* model [43]. This model arose from the observation that many natural systems pass through a cycle of exploitation and growth, then a stable phase in which the structures of the system become increasingly rigid and resistant to change, and then ultimately a collapse and a restructuring. A simple example is the growth of a forest and its periodic clearing through forest fire. The cycle of collapse and restructuring is often necessary to release resources for new growth and new innovation. Panarchy theory suggests that this cycle operates at multiple levels in a system hierarchy at once, with a slower progression at the larger scale, and more rapid cycles of creative destruction at the smaller scale. Management theorist have adopted this model for corporations, to explain how a large corporation can maintain its overall identity and market share, even as units (or product sales) within it grow, stabilize and then collapse.

Panarchy theory is particularly relevant to reasoning about transformational change for sustainability, as it suggests that large-scale system change occurs when change at the smaller scale cascades upwards. However, these upward cascades only trigger broader change when the larger system is at an appropriate point in its own cycle, for example when structural rigidities have set in, causing a loss of resilience. This model therefore offers an opportunity to analyze the broader social and environmental context for innovation in ICT, to help understand the role of such innovations in bringing about transformational change.

C. Tools for Critical Analysis

A third weakness of computational thinking is that is does not encourage critical thinking. This lack of critical thinking is evident in the technical literature, and even more so in stories of ICT in the media, where each new piece of technology is described in terms of its immediate functionality, without any regard to its impact on the broader divisions in society, or its lifetime impact on the environment.

That is not to say that computational thinkers lack the tools to evaluate whether a technological solution will work as planned. However, CT lacks the tools to ask deeper questions about truth and meaning. Humans increasingly define ourselves in relation to the technology we use in our lives [44]. Such technology can have a liberating effect for some people, even as it disempowers others. In the social sciences, critical theory asks questions about how relationships of power are created and maintained in society, and how the tools that mediate social interactions affect these relationships. In particular, critical theory asks questions about who has agency in a given situation, for example, who has the power to create or prevent change. An approach to technology that ignores these issues cannot support anything but a very shallow discussion of sustainability, because transformational approaches to sustainability require some re-alignment of political power. Hence, some form of critical theory is needed as part of the conceptual toolkit of practitioners of ICT for sustainability.

Some approaches to systems thinking also lack such a critical perspective, but there is a long tradition within some branches of the field to apply this kind of analysis, for example, in Soft Systems Methodology [45], and more recently, in Flood's Critical Systems Thinking [46].

Systems thinking lends itself naturally to a critical approach, because it encourages the idea that any system can be seen as a component in a larger system. Hence, a decision to treat a set of phenomena as a system must be accompanied by an appreciation of systemic effects in both the broader containing system and within individual subsystems. Furthermore, any interesting system is likely to be sufficiently complex that different people will describe it in different ways. Weinberg calls this the *Principle of Complementarity* [47]. In theory, it should be possible for two people who have complementary views of a system to make further observations to reconcile their views completely. However, in practice, it is frequently impossible to make sufficient observation of a complex system to completely reconcile all discrepancies between their views; hence, disagreement over the nature of a system may be inevitable.

It is also clear that our mental models of a system constrain our ability to engage with it. While the term cybernetics is often used to describe our ability to interact with (and control) system-level behaviours, *second-order cybernetics* refers to the idea that our observations of a system and our understanding of it are related, and they interact in complex ways [48]. Our understanding of a system determines what type of observations we are likely to make of it, and those observations in turn shape our understanding. We can then conceive of a system for observing and modelling systems, and explore how much control we (as thinkers) have over *this* system.

Some systems thinkers go so far as to define a system as a *way of viewing* the world, rather than as something that can be said to exist objectively, to emphasize the fact that deciding to draw a boundary around something and call it a system represents an explicit analytical choice. This idea forms the basis for Midgley's *Boundary Critique* [49]. The decision for where to draw this boundary reflects the subjective concerns of the observer, and hence it privileges those concerns, and marginalizes other concerns. Boundary critique is the process of exploring the reasons for drawing a systems boundary in a certain way, and in particular, to expose how boundary drawing can be used to exercise power over those who are marginalized by a particular choice. Such an analysis is needed for sustainability, where boundary critique might be used to ask questions about whose interests are served when we treat sustainability as a problem of individual behaviour (and how to re-shape it), rather than as the collective responsibility of a community or a corporation.

V. Teaching Systems Thinking

Sterman argues that if we are to build a coherent science of sustainability, it must be based on concepts such as those described in the previous section [11]. However, many of these concepts are subtle and frequently counter-intuitive. One of the reasons that systems thinking is not more widely taught and practiced is because it is usually presented as a set of abstract concepts far removed from everyday practice, just as I have described them above. For example, while we can treat the business context of a firm in terms of a set of systems, the insights from doing this do not typically provide an immediate set of useable techniques for managers to put into practice [50]. Similarly, systems thinking ideas often do not connect in any strong way with the praxis of technology design and deployment. So even those who are trained in systems thinking (and there are still relatively few) are often unable to apply these ideas in a meaningful way.

So, on the one hand, I have argued that systems thinking provides a necessary conceptual toolkit for understanding sustainability, and an important complement to computational thinking. On the other hand, systems thinking has had little impact in the past, because the concepts are hard to teach and hard to learn [50].

Another barrier to teaching and practicing systems thinking is that it does not fit neatly into any existing discipline within the structure of higher education. This limits both the further development of a body of theory and practice, as research on systems thinking is diffuse within academic institutions, and it limits students' exposure to the ideas, because it does not fit neatly into existing degree structures.

To tackle both of these issues, we are exploring new ways of teaching systems thinking via a set of hands-on activities, at various levels within a traditional computer science department. In contrast to other initiatives in computer science education that use online games and simulations as a way of stimulating student's interest, we adopt a deliberately low tech and physical approach. Most of these games involve students interacting in the classroom (and sometimes outside!) far from their laptops and smartphones. Our goal is to create a space in the class to step back from our everyday experience of technology, and consider the broader systems in which that experience operates.

The games we have used are primarily drawn from "*The Systems Thinking Playbook*" [51], along with a further volume of games aimed specifically at getting students to think about climate change via systems thinking [52]. We also use additional systems thinking games from other sources, such as the Beer Game [53].

We use these games to structure a series of seminars. Each seminar typically starts with one of the games, and leads into a structured debriefing in which students collectively tease out the systems concepts exhibited by the game, along with their own reactions to the overall behaviour of the system. A frequent lesson is that a few simple rules to a game sets up a system structure in which individual action (and choice of action) often has much less impact than the students expect. Instead, the overall behaviours arise from the system structure, and each game tends to follow a similar pattern, no matter which group of students play it. If the class is large enough, we sometimes divide the students into two separate groups, and get each group to play the game while the other group observes. Students are often surprised to note that the same kind of behaviours recur in both instances of the game, and also that their experience of these behaviours is vastly different, between playing versus observing.

VI. Integrating Systems Thinking into the ICT Curriculum

I have applied this approach to teaching systems thinking in a number of different courses over the last few years:

- A first year undergraduate inter-disciplinary course on climate informatics, which used the systems games as a route into understanding the use of simulation models in the earth sciences, and in particular, the way in which climate scientists develop and use global climate models (taught in 2011 and 2012)
- A first year undergraduate course on the social and environmental impacts of the internet, in which systems games were used to provide a basis for an inter-disciplinary analysis of whether, on balance, the internet helps us achieve sustainability, or undermines it (taught for the first time in 2013-4)
- A graduate level course on systems thinking for global problems, which explores the research literature on systems thinking in some depth, using the games to explore the concepts described in the readings (taught in 2012, 2013 and 2014).

A full description of the techniques used in these courses, and an evaluation of their effectiveness, is beyond the scope of this paper[3]. However, reaction to the system games has been extremely positive, especially on the courses I have taught multiple times. For these courses, student evaluations of the course are significantly higher than most other courses within the department, with students particularly commenting

[3]I expect to publish a detailed evaluation soon.

on how the systems games reinforced their understanding of the concepts. A common theme in student feedback has been that more students ought to be able to take these courses. Enrolment on the graduate level course has skyrocketed, making it now one of the largest graduate courses in our computer science department.

All three of these courses were targeted at an inter-disciplinary mix of students, and participation from outside of computer science has been greater than from within. Computer science students tend not to select inter-disciplinary courses, partly because the demands of their degree programs offer very little scope for it, and partly because they judge such courses to be outside the scope of their interests. For the future we are exploring ways of integrating systems concepts into the computer science curriculum in other ways. Rather than isolating systems thinking to a separate course, we are exploring how to bring these ideas into core CS courses. For example:

- In introductory programming courses, the use of examples that illustrate simple systems concepts.
- In a data structures course, assignments that have students building simulations of complex systems with emergent behaviour, such as artificial life simulations. Downey's book *Think Complexity* is a data structures course designed in exactly this way [54].
- In a course on social implications of computing, exercises that explore feedback loops in social use of technology, to explore how systems concepts shape our interaction with technology, and constrain the ability of technology to bring about societal change.
- In a systems analysis course, the introduction of concepts such as boundary critique and participatory design.
- Engineering design courses that incorporate full lifecycle analysis as an essential step in any design process.
- Software engineering courses that adopt and contribute to open source projects, to illustrate the dynamics that occur in a broader community project (See for example, the UCOSP project [55]).

VII. Conclusion

In this paper, I argued that the societal transformations needed to achieve sustainability are more often hampered by ICT than they are helped. I identified computational thinking as an important factor, as it tends to push computer professionals towards overly simplistic formulations of complex societal problems, and fosters technological solutionism - a belief that solving these simplified problems will help build a more sustainable world.

Hence, we tend to look for solutions that automate and optimize existing ways of doing things, in preference to seeking more fundamental transformations towards sustainability. The reductionism of computational thinking offers an impoverished approach to dealing with systemic problems such as sustainability, and in the process blinds us to issues such as the social and environmental impacts of ICT.

In contrast, a fuller understanding of the role of ICT for sustainability requires a different kind of thinking, taking into account the emergent properties of complex systems, and the ways in which the dynamics of social systems shape our use of technology within them. I argued that systems thinking provides a useful antidote to the reductionism of computational thinking, and identified three specific contributions systems thinking can make to expanding the conceptual toolkit of computational thinkers, namely: a domain ontology for reasoning about sustainability, a set of theories of how transformational social change occurs, and a set of practices for critical thinking about the social and environmental impacts of technology.

While the techniques of systems thinking are not new, they have so far made little impact in most academic disciplines. I argued that this is partly because existing university structures do not encourage teaching and research into inter-disciplinary ideas, and partly because the ideas are often considered too abstract to be useful. To address this, we are exploring the introduction of systems thinking into computer science courses, via a collection of inter-disciplinary games that offer hands-on experience of the non-linear dynamics of complex systems. These games work well with variety of students, and are especially useful in mixed inter-disciplinary groups, as they overcome barriers that arise from mixing different levels of expertise within the same class. Response to the use of these games in the courses has so far been overwhelmingly positive, although a full evaluation of their effectiveness will be the subject of future work.

Acknowledgment

I would like to thank the anonymous reviewers for detailed and constructive comments on an earlier version of this paper. I would also like to thank all the students who have taken the systems thinking courses, and my colleagues, Miriam Diamond and Pamela Klassen, for many inspiring discussions about these ideas and their application. Funding for this research is provided by NSERC.

References

[1] P. Teehan and M. Kandlikar, "Comparing embodied greenhouse gas emissions of modern computing and electronics products," *Environmental science & technology*, vol. 47, pp. 3997–4003, 2013.

[2] P. A. Wäger, D. J. Lang, D. Wittmer, R. Bleischwitz, and C. Hagelüken, "Towards a more sustainable use of scarce metals: a review of intervention options along the metals life cycle," *Gaia*, vol. 21, no. 4, pp. 300–309, 2012.

[3] B. H. Robinson, "E-waste: an assessment of global production and environmental impacts." *The Science of the Total Environment*, vol. 408, no. 2, pp. 183–91, Dec. 2009.

[4] K. Dickerson, D. Torres, J.-M. Canet, J. Smiciklas, D. Faulkner, C. Bueti, and A. Vassiliev, "Using ICTs to Tackle Climate Change," Global e-Sustainability Initiative (GeSI), Brussels, Belgium, Tech. Rep., 2010.

[5] D. A. Notter, R. Meyer, and H.-J. Althaus, "The Western lifestyle and its long way to sustainability." *Environmental science & technology*, vol. 47, no. 9, pp. 4014–21, May 2013.

[6] J. Guiltinan, "Creative Destruction and Destructive Creations: Environmental Ethics and Planned Obsolescence," *Journal of Business Ethics*, vol. 89, no. S1, pp. 19–28, Aug. 2008.

[7] E. Morozov, *To Save Everything, Click Here: The Folly of Technological Solutionism.* PublicAffairs, 2013.

[8] R. Perkins, "Technological lock-in ," in *Online Encyclopaedia of Ecological Economics*. International Society for Ecological Economics, 2003, no. February, pp. 1–8.

[9] V. Braun and C. Herstatt, "The freedom-fighters: How incumbent corporations are attempting to control user-innovation," *International Journal of Innovation Management (ijim)*, vol. 12, no. 03, pp. 543–572, 2008.

[10] A. Madrigal, "Bruce sterling on why it stopped making sense to talk about 'the internet' in 2012," *The Atlantic*, December 2012.

[11] J. D. Sterman, "Sustaining Sustainability: Creating a Systems Science in a Fragmented Academy and Polarized World," in *Sustainability Science: The Emerging Paradigm and the Urban Environment*, M. P. Weinstein and R. E. Turner, Eds. New York, NY: Springer New York, 2012, pp. 21–58.

[12] J. Wing, "Computational Thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33–35, 2006.

[13] I. Lee, "CSTA Computational Thinking Task Force," 2014.

[14] P. J. Denning, "Beyond computational thinking," *Communications of the ACM*, vol. 52, no. 6, p. 28, Jun. 2009.

[15] E. Jones, "The Trouble with Computational Thinking," 2011. [Online]. Available: http://csta.acm.org/Curriculum/sub/CurrFiles/JonesCTOnePager.pdf

[16] A. Maslow, *The psychology of science: A reconnaissance*. Maurice Bassett Publishing, 1966.

[17] S. Mann, K. Costello, M. Lopez, D. Lopez, and N. Smith, "An ethical basis for sustainability in the worldviews of first year students," in *Proceedings of the 2nd International Conference on ICT for Sustainability (ICT4S'2014)*. Atlantis Press, 2014.

[18] J. A. Goguen, "The Dry and the Wet," in *Information Systems Concepts*, E. Falkenberg, C. Rolland, and E.-S. El-Sayed, Eds. Elsevier North Holland, 1992, pp. 1–17.

[19] S. M. Easterbrook, "Negotiation and the Role of the Requirements Specification," in *Social Dimensions of Systems Engineering: People, processes, policies and software development*, P. Quintas, Ed. London: Ellis Horwood, 1993, no. July 1991, pp. 144–164.

[20] ——, "Resolving Requirements Conflicts with Computer-Supported Negotiation," in *Requirements Engineering: Social and Technical Issues*, M. Jirotka and J. Goguen, Eds. London: Academic Press, 1994, pp. 41–65.

[21] J. Koomey, "Separating Fact from Fiction: A Challenge for the Media [Soapbox]," *IEEE Consumer Electronics Magazine*, vol. 3, no. 1, pp. 9–11, Jan. 2014.

[22] C. S. Holling, "Understanding the Complexity of Economic, Ecological, and Social Systems," *Ecosystems*, vol. 4, no. 5, pp. 390–405, Aug. 2001.

[23] S. M. Manson, "Simplifying complexity: a review of complexity theory," *Geoforum*, vol. 32, no. 3, pp. 405–414, Aug. 2001.

[24] B. H. Walker and D. Salt, *Resilience thinking: sustaining ecosystems and people in a changing world*. Island Press, 2006.

[25] R. T. Hasty, R. C. Garbalosa, V. a. Barbato, P. J. Valdes, D. W. Powers, E. Hernandez, J. S. John, G. Suciu, F. Qureshi, M. Popa-Radu, S. S. Jose, N. Drexler, R. Patankar, J. R. Paz, C. W. King, H. N. Gerber, M. G. Valladares, and A. a. Somji, "Wikipedia vs Peer-Reviewed Medical Literature for Information About the 10 Most Costly Medical Conditions." *The Journal of the American Osteopathic Association*, vol. 114, no. 5, pp. 368–73, May 2014.

[26] H. W. J. Rittel and M. M. Webber, "Dilemmas in a General Theory of Planning," *Policy Sciences*, vol. 4, pp. 155–169, 1973.

[27] H. Daly, *Steady-state economics*. Island Press, 1977.

[28] T. Jackson, *Prosperity without Growth*. Routledge, 2011.

[29] S. Davoudi, "Resilience: A Bridging Concept or a Dead End?" *Planning Theory & Practice*, no. May, pp. 299–333, 2012.

[30] J. Randers, "Global collapseFact or fiction?" *Futures*, vol. 40, no. 10, pp. 853–864, Dec. 2008.

[31] I. Feygina, J. T. Jost, and R. E. Goldsmith, "System justification, the denial of global warming, and the possibility of "system-sanctioned change"." *Personality & social psychology bulletin*, vol. 36, no. 3, pp. 326–38, Mar. 2010.

[32] H. Brooks, "Can Technology Assure Unending Material Progress?" in *Progress and its Discontents*, G. A. Almond, M. Chodorow, and R. H. Pearce, Eds. Berkeley: University of California Press, 1981, pp. 281–300.

[33] S. J. Davis, K. Caldeira, and H. D. Matthews, "Future CO2 emissions and climate change from existing energy infrastructure." *Science*, vol. 329, no. 5997, pp. 1330–3, Sep. 2010.

[34] J. D. Sterman, "Communicating climate change risks in a skeptical world," *Climatic Change*, vol. 108, no. 4, pp. 811–826, Aug. 2011.

[35] The Natural Step, "The four system conditions of a sustainable society," 2014. [Online]. Available: http://www.naturalstep.ca/four-system-conditions

[36] D. H. Meadows, *Thinking in systems: A primer*. Chelsea Green Publishing, 2008.

[37] B. Nuseibeh and S. M. Easterbrook, "Requirements Engineering: A Roadmap," in *The Future of Software Engineering. (Companion volume to the proceedings of the 22nd IEEE/ACM International Conference on Software Engineering (ICSE2000)*, A. C. W. Finkelstein, Ed. Limerick, Ireland: IEEE Computer Society Press, 2000, pp. 35–46.

[38] M. Chitnis, S. Sorrell, A. Druckman, S. K. Firth, and T. Jackson, "Turning lights into flights: Estimating direct and indirect rebound effects for UK households," *Energy Policy*, vol. 55, pp. 234–250, Apr. 2013.

[39] S. Lewandowsky, U. K. H. Ecker, C. M. Seifert, N. Schwarz, and J. Cook, "Misinformation and Its Correction: Continued Influence and Successful Debiasing," *Psychological Science in the Public Interest*, vol. 13, no. 3, pp. 106–131, Sep. 2012.

[40] W. B. Arthur, S. Durlauf, D. Lane, and D. A. Lane, "Introduction: Process and Emergence in the Economy," in *The Economy as an Evolving Complex System II*, W. B. Arthur and S. Durlauf, Eds. Addison-Wesley, 1997.

[41] D. H. Meadows, "Leverage Points: Places to Intervene in a System," The Sustainability Institute, Tech. Rep., 1999.

[42] L. Dai, D. Vorselen, K. S. Korolev, and J. Gore, "Generic Indicators for Loss of Resilience Before a Tipping Point Leading to Population Collapse," *Science*, vol. 336, no. 6085, pp. 1175–1177, May 2012.

[43] L. Gundersson and C. S. Holling, *Panarchy: Understanding Transformations In Human And Natural Systems*. Island Press, 2002.

[44] T. Winograd and F. Flores, *Understanding Computers and Cognition: A New Foundation for Design*, ser. Language and being. Ablex Publishing Corporation, 1986.

[45] P. Checkland, "Soft Systems Methodology : A Thirty Year Retrospective a," *Systems Research and Behavioral Science*, vol. 17, pp. 11–58, 2000.

[46] R. L. Flood, "Liberating Systems Theory: Toward Critical Systems Thinking," *Human Relations*, vol. 43, no. 1, pp. 49–75, Jan. 1990.

[47] G. M. Weinberg, *An Introduction to General Systems Theory*. Dorset House, 2001.

[48] F. Heylighen and C. Joslyn, "Cybernetics and Second-Order Cybernetics," in *Encyclopedia of Phyiscal Science and Technology*, R. A. Meyers, Ed. Academic Press, 2001.

[49] G. Midgley, I. Munlo, and M. Brown, "The theory and practice of boundary critique : developing housing services for older people," *The Journal of the Operational Research Society*, vol. 49, no. 5, pp. 467–478, 1998.

[50] K. Warren, "Why has feedback systems thinking struggled to influence strategy and policy formulation? Suggestive evidence, explanations and solutions," *Systems Research and Behavioral Science*, vol. 21, no. 4, pp. 331–347, Aug. 2004.

[51] L. Booth Sweeney, *The systems thinking playbook: Exercises to Stretch and Build Learning and Systems Thinking Capabilities*. Chelsea Green Publishing, 2010.

[52] L. B. Sweeney, D. Meadows, and G. M. Mehers, *The Systems Thinking Playbook for Climate Change: A Toolkit for Interactive Learning*. Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ), 2011.

[53] K. Riemer, "The Beer Game Portal," 2012. [Online]. Available: http://www.beergame.org/

[54] A. B. Downey, *Think Complexity*. Green Tea Press, 2011.

[55] E. Stroulia, K. Bauer, M. Craig, K. Reid, and G. Wilson, "Teaching Distributed Software Engineering with UCOSP: The Undergraduate Capstone Open-Source Project," in *Proceedings of the 2011 ACM Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development*. ACM Press, 2011, pp. 20–25.