Emergent engineering: a radical paradigm shift

Mihaela Ulieru*

Adaptive Risk Management (ARM) Laboratory,
Faculty of Computer Science,
University of New Brunswick,
P.O. Box 4400,
Fredericton, NB E3B 5A3, Canada
E-mail: ulieru@unb.ca
*Corresponding author

René Doursat

Institut des Systèmes Complexes,
CNRS and CREA,
Ecole Polytechnique,
57–59, rue Lhomond,
Paris 75005, France
E-mail: rene.doursat@polytechnique.edu

Abstract: We shed light on the disruptive advances brought by the ubiquity of computing and communication environments, which link devices and people in unprecedented ways into a new kind of techno–social systems and infrastructures recently named 'cyber-physical ecosystems' (CPE). While pointing to fundamental biases that prevent the traditional engineering school of thought from coping with the magnitude in scale and complexity of these new technological developments, we attempt to lay out the foundation for a new way of thinking about systems design, referred to as *emergent engineering*. One major characteristic of CPE is that, given their very nature, they cannot be *a priori* defined but rather *emerge* from the interactions among a myriad of elementary components. We show how this emergence can be guided by balancing positive and negative feedback, which tunes the growth of new configurations and adapts the system to sharp and unexpected changes. Rather than attempting to design the system as a whole, the *components* of the system are endowed with *capabilities* of dynamic self-assembly, disassembly and re-assembly to enable 'evolve-ability'. As paradoxical as it may seem to the classically trained systems engineer, this new attitude of the designer as an 'enabler' (vs. 'dictator' of a system's blueprint) allows the system to seamlessly adapt its development and evolve to meet dynamic goals and unexpected situations in an anticipative manner – an impossible feat under the traditional approach. To the extent that it produces new functionality, the proposed method enables a system to evolve via its ability of pervasive adaptation. Emergent engineering lies at a boundary where theoretical discovery meets systems engineering, computing and communications into a new convergent science of complex systems design. It currently transforms systems and software engineering by embracing various highly interdisciplinary perspectives.

Keywords: complex systems; emergent engineering; emergence and self-organisation; CPE; cyber-physical ecosystems; developmental biology; co-evolution.

Reference to this paper should be made as follows: Ulieru, M. and Doursat, R. (2011) 'Emergent engineering: a radical paradigm shift', *Int. J. Autonomous and Adaptive Communications Systems*, Vol. 4, No. 1, pp.39–60.

Biographical notes: Mihaela Ulieru is the Canada Research Chair in Adaptive Information Infrastructures for the eSociety and Director of the Adaptive Risk Management Lab, conducting research in complex networks as control paradigm for complex systems at the University of New Brunswick. She currently champions the area of emergent engineering and its application to emergency response management and networked enabled operations. She is the founder and leader of several international research consortia, and was appointed on several national and international advisory boards and review panels, among which the Scientific Council of the EU, NCE Intelligent Manufacturing (I*PROMS), the EU FP7 Proactive Initiative on Pervasive Adaptation (PERADA), Australia's Digital Ecosystems and Business Intelligence Institute, and Singapore A*STAR. In 2007, she was appointed by the Minister of Industry as a member of the Government of Canada's Science Technology and Innovation Council.

René Doursat is Director of the Complex Systems Institute, Paris Ile-de-France (ISC-PIF) and Full Member of CREA, the research centre in cognitive science and self-organisation at the Ecole Polytechnique, Paris. Previously, he was a Visiting Assistant Professor in computer science at the University of Nevada, Reno. An alumnus of the Ecole Normale Supérieure, Paris, he came back to academia full-time in 2004, after a segue through San Francisco Bay Area's software industry. His research activities address the computational modelling and simulation of swarm multi-agent systems aimed at a new form of engineering inspired by biological and social complexity – in particular the emergence, dynamics and evolution of heterogeneous architectures. He is the Principal Organiser of the French Complex Systems Summer School in Paris and several international workshops in complex systems science and engineering. He is an Associate Editor of *IEEE Transactions on Neural Networks*, and an expert reviewer or advisor for several journals, grant agencies, award juries and curriculum committees.

1 Introduction

Information and communication technologies (ICT) pervading everyday objects and infrastructures, the future 'Internet of Things' (ITU Internet Reports, 2005) is envisioned to undergo a radical transformation from today's mere communication highway into a vast *hybrid network* seamlessly integrating physical, mobile and static systems to power, control or operate virtually any device, appliance or system/infrastructure. Manipulating the physical world will occur locally, but control and observability will be enabled safely and securely across an overlay network that we broadly refer to as an 'eNetwork'. Such eNetworks will enable the spontaneous creation of collaborative societies of otherwise separate artefacts, referred to as 'cyber-physical ecosystems' (CPE).[1] Their examples range from self-reconfiguring manufacturing plants (Ulieru, 2004) and self-stabilising energy grids to self-deploying emergency taskforces, all relying on a myriad of mobile devices, software agents and human users that would build their own eNetwork on the sole basis of local rules and peer-to-peer communication (Dressler, 2007). In such 'opportunistic ecosystems' (herewith referred to as *eNetworked CPE*) that will make the

Internet of Things, distributed systems at various levels of resolution, ranging from single devices to spaces, departments and enterprises, are brought together into a larger and more *complex* 'system of systems', in which the individual properties or attributes of single systems are dynamically combined to achieve an emergent desired behaviour of the synergetic ecosystem.

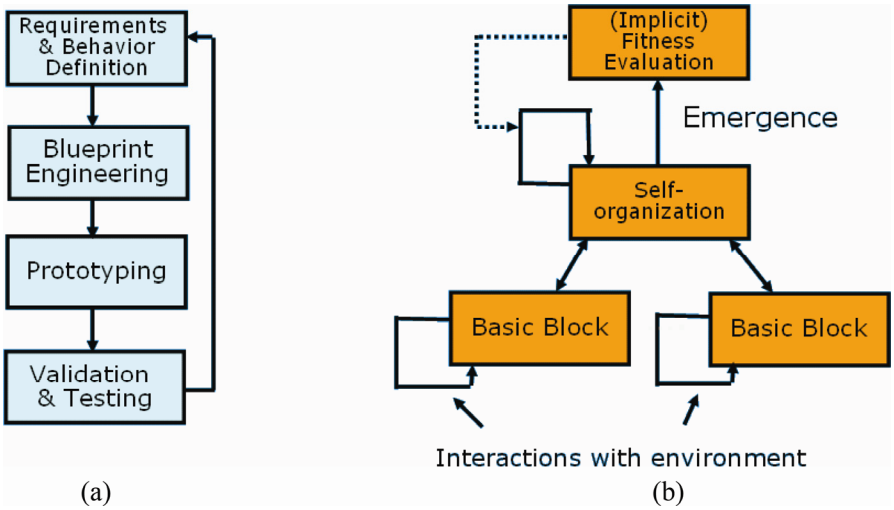
The dramatic progress of CPE technologies is envisioned to reach unanticipated levels of complexity, beyond the boundaries of the disciplines that conceived their components (CPS, 2008). This challenges the traditional engineering school of thought in disruptive ways, given that, by their very nature, CPE cannot be *a priori* defined, but rather *emerge* from the interactions between individual systems' (and people's), interactions facilitated by the eNetworks. This requires to drastically revise the traditional top–down perspective on system design and control, which aimed at imposing order *exogenously*, telling each element of the system what to do at every step through predetermined strategies, and assuming that all possible situations the system might confront are knowledgeable in advance. Instead of fighting it, eNetworked CPE could be managed by 'riding the wave' of their own complexity and rather let systems grow, function and stabilise – even adapt and improve – *endogenously*, in a 'bottom–up' fashion.

2 Towards a new way of thinking about systems design

We address the radical shift of paradigm in systems and software engineering caused by the irruption of ubiquitous computing and communication environments. The accelerated expansion of eNetworks, tightly linking systems and people in unprecedented ways, has enabled a spontaneous and uncontrolled 'bottom–up' emergence of hyper-distributed CPE. Machines, critical infrastructures, softwares and users are now blended at a magnitude and level of complexity that exceeds the traditional 'top–down' engineering mindset. This has puzzled systems and software engineers for some time now and started a worldwide revolution (IT Revolutions, 2008) that aims at a new way of thinking about such complex systems. The new quest is to find appropriate methods to manage the magnitude of scale and complexity of large CPE.

One major characteristic of large interdependent CPE is that, by their very nature, *they cannot be a priori defined* but rather emerge from the interactions between individual machines and people, facilitated by eNetworked communication. Recent attempts to understand and handle these new types of networks point to an alternative school of thought in systems and software engineering, questioning the main stream in disruptive ways. Instead of defining the system and its performance requirements in advance, following a top–down hierarchical thinking (Figure 1(a), inspired by Carreras et al., 2009), the engineer must rather act as a *facilitator* to support and guide the complex system through its process of 'self-design', which generates organisational structure from the bottom–up interactions among a myriad of elementary components (Figure 1(b)). As paradoxical as it may seem to the classically trained systems engineer, this new attitude of the engineer as *enabler* (vs. 'dictator' of a system's blueprint) allows the system to seamlessly adapt its development and evolve to meet dynamic goals and unexpected situations in an anticipative manner – an impossible feat under the traditional approach.

Figure 1 The radical shift in design paradigm: (a) top–down design and (b) bottom–up 'design by emergence' (see online version for colours)



Building on these trends, time is ripe to capitalise on the recent advances in systems engineering, computing and communications, and develop a new, convergent science of complex systems design. The significant difficulty of this pursuit is that it lies at the junction between multiple disciplines: engineering (dynamical systems and control), communications (networks), computer science (agent-based modelling and simulation (ABMS)), physics (statistical mechanics) and biology (self-organisation in morphogenesis, homeostasis and evolution). We need to continue building upon the latest paradigms, through which the new school of thought is currently transforming systems and software engineering, towards a global approach embracing various perspectives from all the above disciplines. We propose to call this unified theoretical effort emergent engineering.

One major mandate of the new school of thought is to formulate and define the concepts of emergent engineering from this radically new, interdisciplinary perspective, as suggested in Lee (2007):

> "Today's computing and networking technologies, however, may have properties that unnecessarily impede progress towards these applications…Many of these applications may not be achievable without substantial changes in the core abstractions…To realize the full potential of Cyber-Physical Systems, we will have to rebuild computing and networking abstractions. These abstractions will have to embrace physical dynamics and computation in a unified way".

This new school of thought encompasses trends in computing and communications as well as networks. In this paper, we attempt to lay out the basis for new concepts and abstractions able to contribute to the development of emergent engineering. Using the paradigms of complexity science, we rephrase the classical concepts of engineering design and systems control respectively, in terms of *developmental emergence*, *adaptation* and *evolvability* found in natural systems to propose a breakthrough approach to the *architecting* and *control* of future eNetworked CPE. We proceed by identifying and

responding to several fundamental biases of traditional engineering in Section 3, and illustrate these new abstractions on a model of self-made network that we propose in Section 4.

3 Fundamental biases carried on from the traditional engineering school

3.1 Traditional engineering requires a system to be well defined

Generally, engineering is about the design of bounded, static systems that can be clearly and completely defined around specific operating points or regions. As systems that continuously adapt and evolve in spontaneous, uncontrollable dynamics, eNetworked CPE cannot be predefined by the designer, be well defined itself. What characterises such large-scale complex systems with unpredictable dynamics is that *non-trivial, large-scale order can be produced by simple processes involving interactions operating locally on simple agents or components*. For such systems – termed emergent holarchies in Ulieru (2004) – 'becoming' is 'being' (Minai et al., 2006). This stands in sharp contrast to the classical paradigm in engineering with its clear distinction between the design and production phase, on the one hand, and the functional phase, on the other hand. Even systems usually considered to be 'adaptive' (such as adaptive controllers or neural networks) follow this two-phase paradigm, allowing adaptation only in the superficial sense of parameter adjustment – whereas complex systems change not only their parameters but also their fundamental structures and processes. This is the essence of the paradigm shift followed by the new school of thought, and the motivation of our work. As stated in Carreras et al. (2007), Lee (2007), and Alderson and Doyle (2009), we need to *design for emergence*, that is, for systems that fundamentally and continually adapt and evolve.

As both a system and an evolving concept at the same time, 'evolution' for eNetworked CPE should not only be construed as a method to optimise the system but more importantly as an *intrinsic property of the system* to be designed (Carreras et al., 2007). Most of complex systems engineering research has focused so far on *specific domains* such as multi-agent systems (Ulieru, 2004), collective robotics and swarms (Gross et al., 2006), and networks (Newman, 2006). However, clues towards a *general strategy* come from the latest insights into developmental biology (Kauffman, 2008), where evolution's profound success is supported by the meta-attribute of *evolvability* as the ability of the configuration space (in this case, the space of genotypes or phenotypes) to produce an endless supply of viable configurations with remarkably few obvious dead ends. Emergent engineering promotes 'evolve-ability' (as per Carreras et al., 2007) as a new paradigm for designing systems capable of evolving towards dynamically changing goals by continuously adapting to unexpected situations without human intervention (Marzano and Aarts, 2003).

Another fundamental insight provided by emergent engineering is that *highly complex functional systems*[2] *can only arise through evolutionary processes of selection in the context of actual tasks*. This fundamentally contrasts with ongoing efforts to design large real-time response systems through specification followed by implementation, which is still the case of even today's distributed systems, applications and techniques involved in multi-agent systems, service-oriented architectures or Web 2.0 and semantic Web – a lingering problem that, for example, 'organic computing' is also trying to address

(Würtz, 2008). Typically, these approaches stem from the traditional top–down design, which 'hardwires' adaptability into the system's blueprint. The blueprint is being while designed in a top–down fashion, thus is fixed, impossible to adapt or change itself (Carreras et al., 2007), designers assume a fixed set of scenarios, decide on a limited range of operating conditions and then build a system that is optimised (in terms of performance) for the chosen applications. Moving farther away from direct design and from the system's profuse details, emergent engineering (EE, 2002), looks rather for the *generic conditions* that will produce those details without dictating them, through a process of developmental and evolutionary 'meta-design'.

As we will attempt to demonstrate through our model in Section 4, emergent engineering endows a CPE with an ability to evolve through a bottom–up *design-by-emergence* approach. Our approach suggests that, rather than attempting to carefully define the system as a whole, efforts should be invested in carefully designing the *components* of the system and endow them with *capabilities* of dynamic self-assembly, disassembly, and re-assembly, in order to enable 'evolve-ability'. Thus, rather than improving the design of a given architecture, the new challenge is to create the premises that can support the *self-design of a whole family of possible architectures*, guided by their intrinsic assembly laws and the extrinsic environmental conditions. As in a jigsaw-puzzle metaphor of system assembly, a component represents a piece of the puzzle, while its binding affinities with other components are embodied in the 'shape' of this piece. At any instant, the system-puzzle finds itself in a certain state, corresponding to a particular compatible arrangement of its pieces. Complex self-assembling systems are multifaceted puzzles: the fit between components is approximate or flexible; component shapes are not unique, allowing for many permutations and equivalent binding configurations; and no one moves the pieces. Rather, old bindings undo themselves and new ones appear, thus seamlessly reconfiguring the system as a function of the ever-evolving environmental circumstances.

The proposed paradigm shift fundamentally challenges the structured and predefined design paradigm of traditional engineering, which envisions each piece as having a predetermined place and functionality in the overall system, crafted for a predetermined scope. Although this radical shift in systems thinking (Boardman and Sauser, 2007) brings unease to the mainstream engineering community at large, it is so far the only path to approach system design for the large-scale eNetworked CPE that are about to shape our world's trajectory in unprecedented ways (IT Revolutions, 2008). Emergent engineering enables the creation of new dynamics of large-scale systems and infrastructures, as well as new methods for managing the complex dynamics of unpredictable complex situations (EE, 2002).

3.2 Traditional engineering requires a system's performance to be specified

Traditional engineering design relies upon a clear definition of the system's *performance* based on the assumption that the system is itself clearly definable. In that context, new and surprising behaviour is construed as anything that falls outside of the system's known or predetermined behaviour and regarded as highly undesirable. Designers assume and predict a finite and fixed set of scenarios, decide on a limited range of operating conditions and then build a system that is optimised (in terms of performance) for the chosen applications (Carreras et al., 2007).

However, as much as one would want, it is not possible to predefine performance criteria for an evolving complex system exhibiting unpredictable emergent behaviour that defies cause-effect behaviour. Here, the performance is rather measured by the *ability of the system to adapt* and accommodate sharp (internal or external) disturbances and dramatically changing operating conditions, while maintaining functionality. Emergent engineering suggests an innovative and original way to address this very difficult problem, namely by regarding performance as *an emergent property* of the adaptive system and designing a controller capable to co-evolve with the adaptive system to seamlessly accommodate such sharp changes in the emergent system behaviour. The system's performance is measured by the system's ability to stabilise quickly around a new operating point far from equilibrium, which emerges from the sharp shift in environmental conditions. This is illustrated by the methodology presented in Section 4. A dynamic fitness function that emerges while the system abruptly adapts to sharp changes in operating conditions is 'tuned' via negative feedback to stabilise the system's growth around a 'most desired' operating point (e.g. created by an attractor in our example of Section 4). To accomplish this, the view of systems' control has to undergo a radical shift to accommodate the paradoxical concept of 'controlling emerging systems'.

The traditional view of control engineering is that the controller is a separate entity that monitors and affects the main system, generally by the feedback from its output variables onto its input variables (Isermann, 1996). In the paradigm shift towards emergent engineering, this system/controller pair becomes fragmented into a myriad of micro-system/micro-controller pairs (represented in our model as simple agents and their individual rules; see also Müller-Schloer and Sick, 2008). Rather than attempting to stabilise the whole complex system in a centralised manner, the emergent controller is implemented in the form of generic control mechanisms located in every component of the complex system. In most typical examples of complex systems, such as pattern formation (e.g. Gierer and Meinhardt, 1972), swarm intelligence (e.g. Bonabeau et al., 1999) or collective motion (e.g. Grégoire and Chaté, 2004), agent rules can be decomposed into two parts: a *positive feedback* that amplifies small local fluctuations in the micro-system, and a *negative feedback* that dampens or corrects the agent's response, and tunes its behaviour more finely (micro-controller). For example, insect colonies provide examples of positive feedback (Bonabeau et al., 1999): ants deposit more pheromone where there is already enough, and termite brings more pellets of soil where there is already a heap of soil. Starting from small initial fluctuations, positive-feedback agent behaviour generally creates a single large homogeneous cluster characterised by some increasing quantity (concentration, size, etc.). More interesting structures can then emerge and be stabilised by adding negative feedback. For example, in collective motion (Grégoire and Chaté, 2004), a bird follows the flock by continuously readjusting its speed and orientation. Each agent corrects small differences by sensing neighbouring agents, and the collectivity converges, albeit temporarily, to a stable trajectory (i.e. the appropriate action plan). Thus, at the emergent level, the tendency of positive feedback is to *create* new mesoscopic or macroscopic structures, while negative feedback tends to *stabilise* them (Grobbelaar and Ulieru, 2007). In other words, bottom–up growth is guided through positive feedback (implemented in the individual rules of the components) while top–down inhibition is regulated by negative feedback (implemented through overall CPE system policies), stopping the growth when it goes outside desired regions. With this, the fitness measure is dynamically attuned via top–down negative feedback to enable the system's adaptation to sharp and unexpected changes in the

environment. These changes, in turn, can only be accommodated by letting the system's components self-organise from the bottom–up, in order to let the system adapt and co-evolve with the dynamic environment.

3.3 Traditional engineering considers complex systems' emergence as an undesirable 'threat'

What traditional engineering fears most is the ability of complex systems to exhibit *emergence*, often assimilated with unwanted behaviour. Surely, goes the quip, one would not want an aircraft to become too creative in mid-flight. Typical questions concern how we can understand such systems and how we can have confidence in the results being produced. Indeed, when starting from such a premise, large collections of autonomously interoperating agents do not appear to be the proper way to address future applications at first sight. Yet, this reasoning is at odds with the striking properties of homeostasis and adaptation reliably displayed again and again by natural systems, from geophysical to biological processes – and life itself, which evolved from emergence (Kauffman, 2008). Instead of aiming to transform all existing and already well-performing systems developed by the solid traditional school into complex systems, emergent engineering addresses the yet unmet design needs of the immense range of yet unaddressed application domains, mostly CPE or domains where the traditional approach failed (CNIP, 2006; Dondossola and Lamquet, 2006; Dunn and Mauer, 2006; IST, 2006; SCADA, 2006).

Taking a closer look at how the internet has evolved into today's complicated network, prone to many pitfalls (Willinger and Doyle, 2002), one notices that the classical engineering paradigm has in fact led to a spiral of increasing complexity characterised by continuous 'patching'. The purpose was to suppress unwanted sensitivities or vulnerabilities – and thereby increase the system's robustness – while taking advantage of new opportunities for increased productivity, performance or throughput. However, the result is far from what we need and is able to achieve from the promises of eNetworks as controllers of large-scale, dynamic and continuously evolving CPE. This is because classical engineering designers aim for robustness at the design stage by seeking to find the right combination of parameter values that keep the system under ideal functioning conditions – something impossible to do for emergent complex systems. The robustness of complex systems goes far beyond optimal settings of a system's parameters, and reaches deep into their *underlying structural properties* that have a major effect on their functionality, dynamics, robustness and fragility (Alderson and Doyle, 2009). In response to this need, emergent engineering enables *robustness-by-structure* achieved by appropriately designing the *interactions* among the system's elementary components (EE, 2002).

Our purpose is to *guide* the emergent behaviour of large-scale eNetworked CPE in such a way that they reach desired performance. These systems can be construed as '(eco)systems of systems' at multiple scales (Ulieru, 2004). They consist of smaller module-systems, component-systems, etc., whose individual properties or attributes dynamically combine to achieve an emergent desired behaviour at the global synergetic level. For such systems, the question is not whether emergence is a good thing or not, but rather how to influence a global behaviour that *necessarily* emerges from the multitude of interactions. The essence of the emergent engineering paradigm is ultimately to find ways to design the controllers for these large-scale eNetworked systems in order to stabilise

their emergent behaviour around desired performance. The whole eNetwork can itself be envisioned as a globally evolving controller, managing the performance of a complex system to be controlled (Grobbelaar and Ulieru, 2007), for example, to use it to stabilise the power grid in case of a blackout or to grow barriers to attacks in a complex crisis and emergency management scenario (Ulieru, 2008).

3.4 Traditional engineering approaches distributed systems design in a top–down centralised manner

The traditional engineering school of thought also induces significant biases when it comes to the more recent and ongoing research in multi-agent, service-oriented and large-scale distributed systems. To better understand this bias, one can broadly categorise the discipline of *distributed intelligent systems* into two families, which we refer to as 'service-oriented agents' and 'simple agents'.

On the one hand, service-oriented agents (e.g. Wooldridge, 2002) come with a huge luggage of semantics and reasoning, which makes them 'intelligent' individually but forces the system developer to design the architecture of their interactions in a deterministic manner, and clearly specify each module from top–down (Figure 1(a)). Distributed service-oriented systems come from a historical trend in software engineering and artificial intelligence that has been gradually replacing big monolithic programmes by clean architectural principles based on layers, modules, objects, etc., that communicate via application programming interfaces (API) (e.g. Tanenbaum and van Steen, 2002). It was realised that disentangling and removing cycles from the graph of function calls allows to group functions into code 'parts', thereby fixing, upgrading or replacing these parts independently from each other, without having to rewrite the rest. Service-oriented systems emphasise the role of software agents as *proxies* representing users or other physical entities and their interests (information-searching internet agents, price-bidding electronic brokers, device-monitoring automation agents, etc.). Here, agents try to satisfy goals under the constraints created by the other agents and their environment.

On the other hand, the alternative 'simple-agent' paradigm is more appropriate to the modelling of CPE as complex adaptive systems (CAS) (Levin, 2003) using ABMS (Macal and North, 2006). They enable a *collective intelligence* operating across multitudes of components at various scales that interact intensively with each other. CAS agents are typically expressed with simpler semantics (Holland, 1998) and are able to produce collective intelligence from their interactions. Agent behaviour can be derived from statistical models and input information (Newman 2006; North and Macal, 2007). Historically, ABMS represents the perspective of social sciences and discrete mathematics, rather than engineering. It arose from the need to model systems that were too complex for analytical descriptions, such as social interactions and the economy (Terranova, 2004). Helped by the rise of computing power, it soon became a practical tool in many other scientific disciplines, such as ecology, biology and physics. Most of ABMS is based on a combination of three types of topologies (Macal and North, 2006): fixed grids such as square pixels, arbitrary networks with long-range connections and 2D or 3D Euclidean space supporting irregular lattices of mobile agents with nearest-neighbour interactions. In contrast to service-oriented multi-agent systems, ABMS rather stresses the *social* interactions among agents towards a collective emergent behaviour with a higher purpose that cannot be identified in the behaviour of the individual parts at a particular scale of observation.

To summarise crudely, the MAS involves a limited number of heavy-weight (code-laden), individual, intelligent agents that perform complex functions, while ABMS tends to rely on many light-weight (few rules), simpler agents that are highly interactive to generate collective intelligence. Emergent engineering explores the links between agents and large-scale distributed systems based on simple agents, along the lines of the ABMS paradigm. Agent properties must be able to meet the management and coordination needs of safety-critical interconnected systems and infrastructures fuelled by inexpensive and ubiquitous sensing, communications and computation. Towards this goal, emergent engineering proposes to construe agents as 'simple', following the seminal works of Holland (1998), Kauffman (2000) and the more recent advances in ICT eNetworks (Carreras et al., 2007; North and Macal, 2007). CPE technologies are envisioned to dramatically evolve over the next years. New properties, issues, interdependencies and vulnerabilities will occur that cannot be envisioned today. To avoid today's solutions becoming tomorrow's problems, a primary requirement for the design of eNetworked CPE is to embed now in their fabric the faculty of 'evolve-ability' mentioned above, that is, the ability of a system to seamlessly accommodate unexpected (either gradual or abrupt) changes by developing new characteristics or properties that the system did not display previously (Carreras et al., 2007).

4 An abstract model of self-made network

In emergent engineering, architecting is done without a global architect. It relies entirely on defining the *basic cells* and the mechanisms by which these cells are able to create reliable architectural components. In this part, we present an abstract model of self-made network based on this idea. It radically departs from service-oriented architectures, in which architectural modules are predefined in a top–down fashion, because it lets architectures grow and evolve from the bottom–up interaction between components. We offer here a methodological framework for *micro-architecting* these elementary components or 'cells', such that they are capable of collectively generating a desired behaviour by emergence, and tuning the dynamic adaptation of the CPE to gradual or abrupt changes in performance requirements and environmental conditions. In doing so, we are seeking generic methods for the design of local interactions that lead, via self-organisation, to a global behaviour while guiding the system towards desired (yet dynamically adapting and evolving) performance criteria.

The few sections of this section present a condensed overview of preliminary results obtained from a new, original model of autonomous network dynamics. We show a model of network *development* and *evolution* that is inspired by the biological development and evolution of organisms and, in this sense, belongs to a class of *artificial embryogeny* (AE) systems (Bentley and Kumar, 1999; Stanley and Miikkulainen, 2003). AE systems are a particular case of evolutionary computation in which the mapping from genotype to phenotype is only *indirect* as it is realised through a complex developmental stage. This is also called an evolutionary developmental or 'evo-devo' approach. Instead of coding directly for macroscopic features of the phenotype (the system), the parameters of the genotype code for microscopic features of the cells are coded (the components), that is, their abilities to communicate, their propensity for motion and their affinities for

assembly with other cells. Like biological cells, nodes in a self-constructing network share the same genotype, that is, the same set of rules. Imitating cell division, differentiation and self-positioning, a node spawns other nodes, follows its own execution path (within the common programme) – which may diverge from its neighbours depending on its position – and creates specific links with other nodes according to this fate.

In eNetworked CPEs, nodes can represent human agents who carry personal digital assistant (PDA) devices with wireless and peer-to-peer connectivity. The self-assembly programme includes routines for the exchange of messages and the dynamical creation or removal of links. It relies on a combination of 'ports' and internal state variables derived from discrete 'gradients'. Ports and gradients guide the new nodes to specific attachment locations in the developing network. As the network expands and node positions change, nodes adapt by switching on or off different subsets of the common set of rules – similar to gene activation/inhibition in biological DNA – thus triggering the growth of specific structures such as chains, lattices and more complicated composite topologies.

Compared to other AE models, such as L-systems (Siero et al., 1982), the novelty of our model resides in the fact that it is both context-dependent (heterogeneous) and self-dissimilar (non-repetitive, irregular), and also that it contains microscopic randomness (at the level of nodes) while it is reproducible at the macroscopic level (of the whole graph, that is, the 'phenotype'). It extends and generalises the principles of pattern formation and collective motion found in morphogenesis from 2D/3D shapes (Doursat, 2006, 2008a,b) (Figure 5) to *n*D-graph topologies.

4.1 Growing simple chains

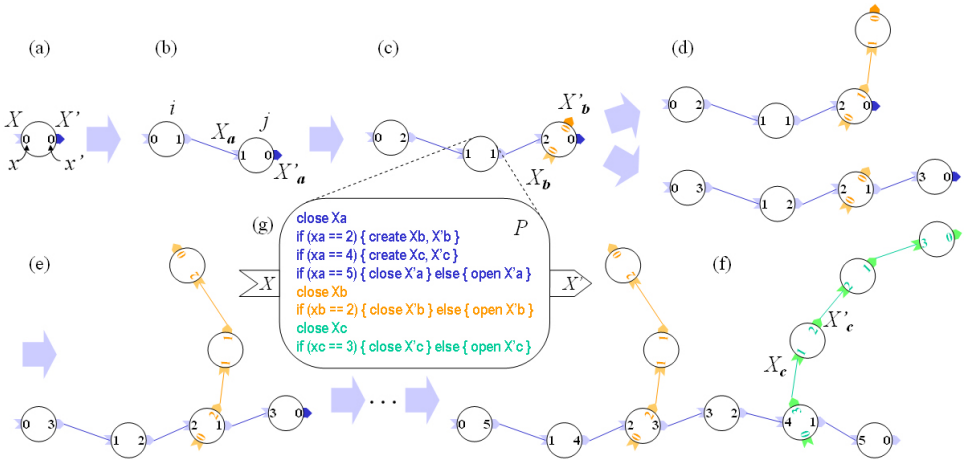
The self-assembling networks envisioned here are composed of dynamical nodes that can carry various pairs of attachment ports (X, X') and corresponding pairs of gradient values (x, x'). Ports can be 'occupied' (linked to other ports on other nodes) or 'free' (not linked), while free ports can be 'open' (available for connections) or 'closed' (disabled). Chains are the simplest self-assembling structures that can be realised with one pair of ports in each node (Figure 2(a)–(c)). New nodes that just arrived in the system's space or nodes that are not yet connected, have both ports open and gradients set to 0. A node i can create a link with another node j only through a pair of complementary open ports, here X and X', with one link per port. As soon as a new link is made, ports are occupied and gradients are immediately updated according to the following rules:

1 a free port always maintains its value at 0 (gradient source)

2 x is sent out through port X' to port X of the neighbour node with an increment of +1 (resp. x', X, X').

Discrete counter increments are also the method of choice for spreading positional information in amorphous and spatial computing systems (e.g. Doursat, 2008b; Nagpal, 2002).

Figure 2 Dynamical and evolvable self-assembly of a network based on programmable nodes. Occupied and closed ports are displayed in light colours, while open ports are displayed in dark colours (see online version for colours)



The purpose of the gradient counters (x, x') is to keep track of the nodes' positions in the chain. This allows, for example, to build chains of a fixed length n by closing ports as soon as $x + x' = n - 1$. It can also create more complicated structures by switching on or off certain attachment rules when certain gradient values have been reached. All nodes carry the same programme (their genotype or 'DNA'), which consists of three main routines: *gradient update* (G), *port management* (P) and *link creation* (L). The gradient update routine G is the generic code that provides nodes with the positional information (x, x') that they need to make further decisions (see propagation and increment rules 1 and 2 above). The port management routine P (illustrated in Figure 2(g)) contains the heart of the logic (the genotype) specific to the construction of a target structure (the phenotype). Routines G and P are executed by the nodes already involved in the network, and prepare the way for new nodes to execute L. Link creation routine L provides the generic logic that prompts new nodes to pick one of the open ports of the network at random to make a new connection.

4.2 Creating modular structures with different gradients

More complicated structures can be developed by composing multiple chains in branching arrangements (Figure 2(d)–(f)). To allow the creation of *modules* with their own identities and local positional information, one can find again inspiration from biology, in particular the concepts of modularity and homology so central in evo-devo (Callebaut and Rasskin-Gutman, 2005). Modules are similar to 'limbs' that have distinct morphologies and geographies. This is modelled here by different coordinate systems based on tags a, b, c, etc. Gradient ports in one part of the system, for example, a chain, are denoted by (X_a, X'_a), while ports in other branches will be (X_b, X'_b), (X_c, X'_c), etc. Accordingly, routine L is amended so that links cannot be created between ports with different tags.

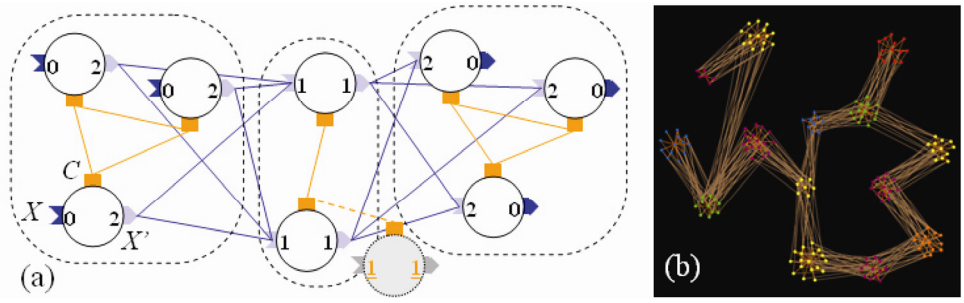
In the simple scenario of Figure 2, only the X'_a port is open in the beginning (Figure 2(a) and (b)). When the third node has attached, another pair of ports (X_b, X'_b) is created on that node and only port X'_b stays open (Figure 2(c)). Note that this particular event is triggered by the positional information carried by the node: in this example, the *P* routine (Figure 2(g)) stipulates that when $x_a = 2$, a node must differentiate into a bifurcation node, that is, create another pair of ports and their corresponding gradient variables. After this event, new nodes can attach to either open port, X'_a or X'_b (Figure 2(d)), that is, either choose to first augment the original chain or its branch. However, the order of node attachment will not modify the final structure. New nodes carry an untagged pair of ports (X, X') and acquire the tag of their first contact. The same 'stop-rule' of chains applies here when the *b* branch reaches length $n_b = 3$, that is, $x_b + x'_b = 2$, closing the only open port X'_b (Figure 2(e)). Independently, another branch *c* grows from the fifth node of chain *a* and stops at $n_c = 4$ nodes, while chain *a* stops at $n_a = 6$ nodes.

4.3 Cluster nodes

In biological development, the position and number of individual cells is very imprecise, while the structures and organs they form are reliably placed. Similarly, programmed network self-assembly could also be irregular at the microscopic level of the nodes, while retaining an orderly arrangement at the higher, 'mesoscopic' levels of *clusters* of nodes. This property of *variability* of an emerging structure, in addition to its fundamental *programmability*, is embodied here by replacing single nodes with clusters (Figure 3). This is done through a special port, *C* (as in 'cluster' or 'clique') that allows multiple nodes with identical gradient coordinates to form random connections with each other. The *C* port represents an extra 'non-linear' dimension added to the pairs of ports (X_a, X'_a), (X_b, X'_b), etc., of any composite structure. Another new feature is that nodes are also allowed to make multiple connections per port, whether *X* or *C*. Thus, in the case of a chain, a new node has two possibilities of attachment: it can either *thicken* or *lengthen* the chain. It either connects to an existing node through *C*, in which case it inherits the coordinates of that node's cluster, or it connects as before via *X* or *X'* ports, in which case it pioneers the creation of a new cluster at one end of the chain and all coordinates are updated according to the usual gradient dynamics. After their first link, new nodes may also establish a few supplementary connections through any of their ports, under the constraint of coordinate consistency (–1 and +1 via ports *X* or *X'*, equal coordinates via port *C*).

Similar to cellular proliferation in morphogenetic tissues and organs, this proliferation of nodes in structured networks introduces redundancy and 'failover' safety. Unlike single-node chains, the failure of one link in a cluster chain does not imply the failure of the whole structure. Yet, while relying on a fluctuating swarm of agents for its robustness, the *overall topology* of a programmed network is still not left to chance but narrowly guided by the genotype of the attachment rules *G*, *P* and *L* to grow the desired structures.

Figure 3 Programmable network topologies, in which the main nodes are in fact composed of clusters of randomly connected sub-nodes. These topologies exhibit both randomness at the microscopic level and precision and reproducibility at the macroscopic level (see online version for colours)



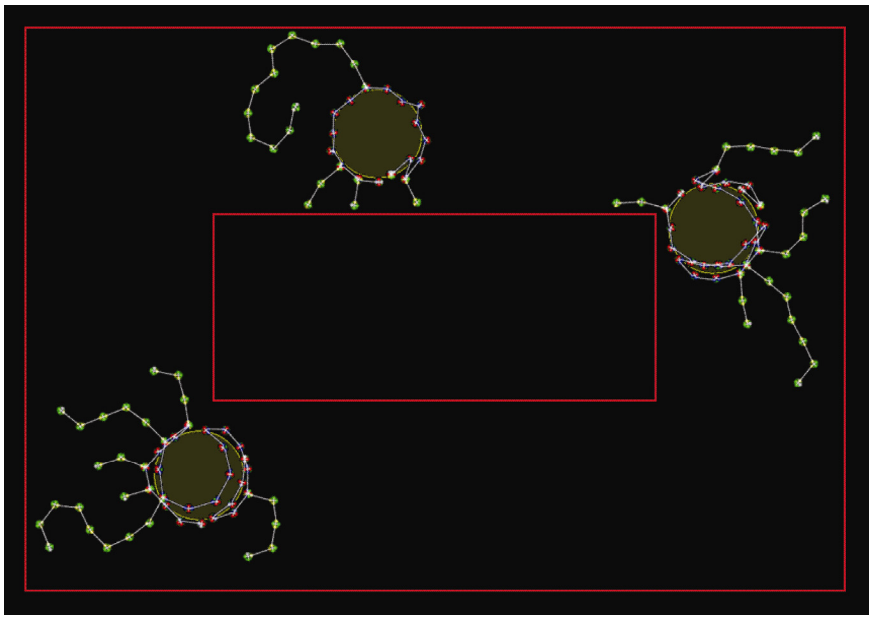
4.4 Further guidelines towards concrete applications

The emergent engineering process described above defines components and their interactions, but the primary challenge is to ensure that the design produces a desired global functionality. The previous section presented abstract mechanisms of self-made networks that have a purely endogenous (i.e. bottom–up) ability to form precise configurations. It established new foundations for the emergence of non-random, programmable patterns exhibiting *intrinsic structures* that are neither repetitive nor imposed by the environment. Starting from these premises, in order to make it applicable to concrete problems, we aim to complete the model with the following features: physical space, developmental adaptation to a dynamic environment, agent functionality, and hierarchical command and control.

Physical space: as mentioned in the Introduction section, most real-world eNetworks combine, to a certain extent, non-spatial graph topologies (e.g. connecting organisations and entities) with Euclidean graph topologies (e.g. connecting people and equipment on the field). The abstract mechanisms of programmed attachment described in Section 4 create purely non-spatial graphs that are displayed in 2D figures only for convenient viewing. Space can then intervene at two levels: by limiting the scope of pre-attachment detection (nodes can connect only to nearby nodes, within a certain radius), and by giving a mechanical meaning to the nodes and links.

Developmental adaptation to a dynamic environment: most importantly, as it is a recurrent theme of this paper, the propensity to create specific network morphologies by programming the nodes must also be *influenced and modified by the environment* in which those formations will function. In Sections 4.1–4.3, node attachment was based on port availability driven only by positional gradient values. This internal dynamics must now interact with the external dynamics of the system's context, via the physical space of the environment, along with all its possible boundary conditions and events occurring unexpectedly. Environmental landmarks can play different roles in the self-structuring process, acting as triggers, attractors or obstacles. Figure 4 gives an example of numerical simulations of self-organised network morphologies – in which nodes execute a program similar to that of Figure 2 – that exhibits a high degree of adaptation to environmental constraints, such as spatial boundary conditions. Each network is based on the same node program (genotype), yet grows differently ('polymorphism' of the phenotype) as it senses its environment.

Figure 4 This numerical simulation of self-organised network morphologies – in which nodes execute a program similar to that of Figure 2 – shows that they can exhibit a high degree of adaptation to environmental constraints, such as spatial boundary conditions. Each network is based on the same node program (genotype), yet grows differently ('polymorphism' of the phenotype) as it senses its environment (e.g. via an anti-collision rule between the nodes and the red walls) (see online version for colours)



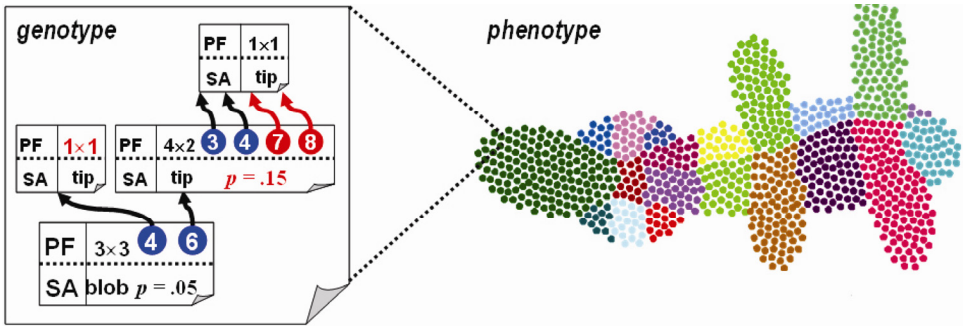
Agent functionality: another important aspect not included in the abstract model is the diversity of functional roles that agents may take on, in addition to their self-assembly capabilities. The model should also mix various *predefined* agent identities before they even further differentiate by gradient position inside the structure. This natural heterogeneity of agents could be reflected in the model by a heterogeneity of ports and gradients, and diversified attachment rules that depend on agent types. This would result in various subnetworks of two kinds: 'intra-category' subnetworks linking agents of the same expertise, and 'inter-category' subnetworks combining agents of different expertise together.

Hierarchical command and control: finally, as discussed in the Introduction section, the adequacy or 'fitness' of the deployed eNetwork to a specific situation, both in its structure and function, might also depend on a two-way communication between the agents and a remaining central supervision. Some CPE cannot exclusively rely on peer-to-peer self-organisation at the local level, and might still need (minimal) monitoring and orchestration at the global level. In this framework, dynamical adaptation to an evolving environment basically can happen at two levels:

1 quick adaptation to local circumstances at the level of the agents *under the same rules of deployment*

2 major changes of strategy at the command level that *change the rules of deployment*.

Figure 5 Genotype and phenotype in artificial embryogeny (see online version for colours)



Source: Doursat (2006, 2008a,b).

High-level command and control action plans would set only the global course of the action, while the low-level implementation details are carried out by individual agent protocols (e.g. real-time positioning). Action plans are compiled down into local rules of attachment and broadcasted to all agents. Thus, the network can adapt to new incidents and episodes of an evolving situation by reprogramming the agents on the fly to create new formations.

In summary, future work can expand the abstract algorithmic rules (gradient update G, port management P and link creation L) to take into account spatial extension, external events, agent diversity and hierarchical command. By implementing these four principles – in addition to intrinsic self-connectivity – self-organised and structured eNetworks could become truly functional and evolvable. This dynamical process would be continuously adjusting to the environment's dynamics, including its unexpected new events and effects. The effectiveness of an eNetwork would depend on how its genotype is designed (i.e. how individual roles are specified through protocols) in such a way as to obtain maximal synergy under the overarching constraints imposed by the phenotype (reflected in network policies; Figure 5). It is this continuous 'balancing act' between individual agent autonomy and overall goals (previously explored in holonic enterprises) that would enable the emergence of effective structures, which grows when and where needed, to face unexpected developing events. This could ensure a continuous adaptation and co-evolution with an environmental dynamics by making an eNetwork CPE (as the controller) 'weave itself' into the situation to control like a nervous system, growing new connections and 'nerves' around important events and locations (Ulieru, 2008).

5 Emergent engineering

5.1 The paradigm shift in a nutshell

To ensure stability and predictability as major desirable systems characteristics, classical engineering often strives to eliminate self-organisation and emergent processes in favour of reductive piece-by-piece design, characteristic of the way *complicated* rather than *complex* systems arise (Alderson and Doyle, 2009). By contrast, the structure of a complex system (Bar-Yam, 2003) is not the result of a historic design process, but a contingent process of evolution (Kicinger, 2004). The primary difference is that systems

designed through the classical engineering process are expected to perform *foreseeable tasks in a bounded environment*, whereas complex systems, either natural (living organisms, insect colonies, ecosystems) or man-made large-scale CPE (communication networks, transportation networks, cities, societies, markets, multinational corporations; Terranova, 2004) are expected to function in *complex, open environments with unforeseeable contingencies*. This requires high adaptability by which the system can evolve novel configurations emerging from clustering its components in new ways.

Optimality and performance: just as traditional engineering seeks optimal solutions, emergent engineering must seek 'optimal' configuration spaces, where near-optimal configurations for an infinite number of as-yet unforeseen circumstances are numerously implicit (Doyle and Csete, 2007). The promise of emergent engineering is, therefore, one of open-ended discovery of new system configurations that can respond to unforeseeable changes, rather than predetermined performance targeted at static environmental conditions.

Utility: in the classical paradigm, utility is assured by the explicit design and testing of the processes that produce the desired functionality and the pathway from component behaviour to system behaviour is clear. This is not the case of engineered complex systems where, by definition, system functionality is emergent and too complex to be described explicitly in terms of component behaviour. New behaviour evolves from the components' interactions, and utility is measured by the degree to which the new behaviour reflects an adequate system adaptation to the environmental changes.

Performance metrics: implicit in most work (Minai et al., 2006) is the notion that complex systems should be judged on their *meta-attributes* such as robustness (Alderson and Doyle, 2009), evolvability (Carreras et al., 2007), adaptability, scalability (Ulieru, 2004), etc., rather than on narrowly defined tasks. However, defining and measuring these properties is still far from being an exact science. Current methods for evaluating engineered systems encompass rigidly specified criteria with well-defined 'correct performance', while we are still lacking metrics to assess the meta-attributes that make a complex system worth its competitive advantages.

Evolution vs. evolvability: traditionally, in engineering, evolutionary methods have been considered to be just another *optimisation technique*, in which human designers create the meta-process of problem specification and interpretation, such as defining a 'fitness function' as a measure of how well the system has improved through evolution. The evolution of large complex systems (called 'evolvability' by Carreras et al., 2007) takes place primarily in their functional environment in which, by enabling the system to adapt to real-world tasks through changes in components and their interactions over time, the system creates new configurations to address abrupt change. Doing so, it evolves new behaviour that was simply not displayed or impossible to display before. In this way, 'evolvability' can encompass 'evolution' because it can create behaviour that would not have been possible before the dramatic adaptation that the system had undergone. Yet, evolvability can be more relaxed and enables the system to only manifest properties that it had but never used, and 'experience itself' through behaviours that were possible but it did not have the opportunity to display before. Due to the particular traits of systems that exhibit emergent behaviour, it is not easy to point to an exact boundary between evolution and adaptation, especially in the case where the system never exhibited certain behaviours, although they were in the plethora of possibilities. What we consider

important is to distinguish between lean adaptation and sharp adjustment of the system's behaviour to accommodate abrupt change – which can also lead to permanent mutation ('true evolution').

Robustness: classical engineering designers seek to find the right combination of parameter values that keep the system under ideal functioning conditions – something impossible to do for emergent complex systems. The robustness of complex systems goes far beyond optimal settings of a system's parameters, and reaches deep into their *underlying structural properties* that have a major effect on their functionality, dynamics, robustness and fragility (Alderson and Doyle, 2009). In response to this need, emergent engineering enables *robustness-by-structure* achieved by appropriately designing the *interactions* among the system's elementary components.

In summary, eNetworked CPE can be modelled as CAS using the ABMS paradigm to build a *collective intelligence*, operating across a multitude of components at various scales that interact intensively with each other. Since CAS agents are relatively simple in their semantics, like cells in natural organisms, the system's intelligence results from their *collective interactions*. Most surprisingly, our deepened understanding of genomics and molecular biology (Kauffman, 2008) has revealed that, at the network and protocol level, cells and organisms are strikingly similar to technological networks, despite having completely different material substrates, construction and evolution dynamics (Doursat, 2008a,b; Doyle and Csete, 2007). Biological agents (cells) carry a set of rules (DNA) that endows them with a repertoire of non-trivial behaviours. Methods to reintroduce a certain dosage of *programmability* inside free self-organisation, in the form of a *developmental genotype* (Figure 5) are explored in the field of artificial development (Bentley and Kumar, 1999; Doursat, 2006, 2008a,b) and amorphous computing (Abelson et al., 1999; Nagpal, 2002). The global behaviour is specified in terms of primitive behaviours at the agent level and this 'programme' is then 'compiled' into a common behavioural specification for all agents, ensuring the emergence of the desired global effect. To date, there is no unified 'complex systems science' or agreed-upon 'complexity theory'. No central dogma or modern synthesis has yet happened for complex systems, as it has for biology. However, a great diversity of related topics and disciplines coexist, and a vast array of mathematical and computational tools were recently proposed (Minai et al., 2006; Newman, 2006). We aim to look at the commonalities across these domains in search for the generic principles of emergent engineering.

5.2 Principles of emergent engineering

From the above considerations, we can envision the following generic principles of emergent engineering:

Architecting from the bottom–up without an architect: a closer look at complex systems (biological or techno–social) reveals that they all consist of a large number of *agents*, which follow a set of micro-instructions or *rules* on how to *search* and *connect* to other agents, *interact* with them over these connections, *change* one's internal state and carry out some specialised *function*. The rules act upon an array of internal *variables – developmental* (dedicated to building the system) and *functional* (dedicated to making the system carry out tasks). The rules can also be modulated by *parameters* that may *evolve* over time, according to a global *fitness* that the system is exhibiting with

respect to its function in the environment. By analogy to biology, our approach considers genetic-like regulation at the agent level to harness large-scale eNetworked CPE. Ultimately, the quest of emergent engineering is to define the blueprint (the DNA structure) of a 'cell' in such a way that architectural components collectively emerge and the eNetwork grows CPE with desired characteristics.

Control without a controller: *Using the eNetwork to control large-scale CPE*: the traditional view of control engineering is that the controller is a separate entity that monitors and affects the main system, generally by feedback from its output variables onto its input variables. It is extremely hard, if not impossible, to control a large-scale eNetworked CPE by building a global-logic, top–down system able to rapidly adapt to changes adequately if each element needs to be instructed about what to do at each step. In the paradigm shift towards emergent engineering, this system/controller pair becomes fragmented into a myriad of micro-system/micro-controller pairs, where a micro-system is a 'cell' and its micro-controller is the subset of rules responsible for stabilising its behaviour. Agent rules can be decomposed into two parts:

1 a positive feedback that amplifies small local fluctuations (micro-system)

2 a negative feedback that dampens or corrects the agent's response, and tunes its behaviour more finely (micro-controller).

At the emergent level, the tendency of the former is to *create* new macroscopic structures, while the latter tends to *stabilise* them. Emergent engineering aims at a methodology to evolve the micro-controller in individual cells, such that eNetworked CPE can deploy emergent desired functionalities.

Thus, the emergent engineering paradigm opens perspectives on how strategies that mimic natural adaptation of highly evolved robust systems can be developed with simple agents: "When one gets a collective behaviour from the bottom-up individual interactions of a multitude of elements, adaptation of the large scale system to unexpected disturbance comes naturally, and only in regions where it is needed" (Levin, 2003).

Co-evolving the CPE with the environmental dynamics: once the basic 'eNetwork DNA' parameters have been set to achieve the CPE growth (architecture) and function (control), the remaining question is how to make the CPE *co-evolve* with the environmental dynamics. After reaching structural maturation on a short deployment time scale, the eNetworked CPE should switch the bulk of its activity from executing the *developmental part* of its genotype (Figures 2(g) and 5) (dynamic architecting, which *positions* the actors within the network so that they can best perform their activity in coalitions or teams) to executing the *functional part* of its genotype (adaptive control obtained by *executing* their roles within the teams to realise the most effective action plans). This can be done by specifying how the genotype (individual agent rules) may vary and how the phenotype (overall CPE network policies that enable the selection of appropriate behaviour) may be selected. The challenge of emergent engineering is to deliver a method to balance the genotype (developmental) and phenotype (functional) parts (Figure 5).

6 Conclusions

In response to the need to manage the complexity of large-scale eNetworked CPE, we proposed a breakthrough in the design of resilient and efficient complex distributed systems that could affect many disciplines in the next decades by radically rethinking *systems engineering*. Emergent engineering attempts to put natural and engineering complex systems within the same discipline – closing the loop between complex systems science and complex systems engineering. In this paradigm, the study of natural complex systems leads to better methods for complex engineered systems while experience with building and manipulating complex engineered systems enhances the understanding of how natural complex systems function. This research will open the door to new inventions enabling the development of solutions crucial for the orderly functioning of society and the economy (EE, 2002). Examples can be found in the resilient deployment of interdependent critical infrastructures and blackout-free optimised power grid, holistic security ecosystems, hazard-free transportation (automotive networks for aerospace and avionics), network-enabled operations (Dorn, 2007), emerging architectures of participation by peer production in organising work, etc. Evolve-able, resilient and efficient CPE unleash a great potential for the seamless integration of yet unthinkable technologies within the fabric of our Planet – thus creating an open environment for far-reaching continuous societal, economic, industrial and technologically sustainable growth. CPE will accommodate both gradual and disruptive developments, whose influence on our lives cannot be fully grasped today, such as the threat of climate change.

Acknowledgements

This collaborative work has been made possible in part by the Scientific Services of the French Embassy in Ottawa, through their funding of Dr. Doursat's visit to Prof. Ulieru's Adaptive Risk Management Laboratory at UNB. We thank Adam MacDonald, MSc student with Prof. Ulieru, for his excellent work in producing the computer simulations using his Fluidix© software package http://www.onezero.ca.

References

Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Knight, T., Jr., Nagpal, R., Rauch, E., Sussman, G. and Weiss, R. (1999) 'Amorphous computing', *MIT Artificial Intelligence Laboratory memo* 1665.

Alderson, D. and Doyle, J.C. (2009) 'Can complexity science support the engineering of critical network infrastructures?' *IEEE Transactions on Systems, Man and Cybernetics, Part A*, Special issue on Engineering Cyber-Physical Ecosystems, July 2009.

Bar-Yam, Y. (2003) *Dynamics of Complex Systems.* Westview Press.

Bentley, P. and Kumar, S. (1999) 'Three ways to grow designs: a comparison of embryogenies for an evolutionary design problem', in W. Banzhaf et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference*. Orlando, Florida: Morgan Kaufmann, Vol. 1, pp.35–43.

Boardman, J. and Sauser, B. (2007) *Systems Thinking: Coping with 21st Century Problems*. CRC Press.

Bonabeau, E., Dorigo, M. and Theraulaz, G. (1999) *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press.

Callebaut, W. and Rasskin-Gutman, D. (2005) (Eds.) *Modularity*. MIT Press.

Carreras, I., Miorandi, D. and Chlamtac, I. (2007) 'From biology to evolve-able ICT systems', *1st International Workshop on eNetworks Cyberengineering, IEEE Systems Man and Cybernetics Conference*, Montreal, Canada, October 7–10, 2007.

Carreras, I., Miorandi, D., Saint-Paul, R. and Chlamtac, I. (2009) 'Bottom–up design patterns and the Energy Web', *IEEE Transactions on Systems, Man and Cybernetics, Part A*, Special issue on Engineering Cyber-Physical Ecosystems, July 2009.

CNIP (2006) *International Workshop on Complex Networks and Infrastructure Protection*, Rome, Italy, March 28–29.

CPS (2008) *Cyber-Physical Systems Summit*, St. Louis, MO, USA, April 24–25.

Dondossola, G. and Lamquet, O. (2006) 'Cyber risk assessment in the electric power industry', *Cigrè Electra Magazine* 224.

Dorn, W. (2007) 'Tools of the trade? Monitoring and surveillance technologies in UN peacekeeping', *Report to the UN Peacekeeping Commission*.

Doursat, R. (2006) 'The growing canvas of biological development: multiscale pattern generation on an expanding lattice of gene regulatory networks', *InterJournal*: *Complex Systems 1809*.

Doursat, R. (2008a) 'Organically grown architectures: creating decentralized, autonomous systems by embryomorphic engineering', in R.P. Würtz (Ed.), *Organic Computing*. Berlin: Springer-Verlag, pp.167–200.

Doursat, R. (2008b) 'Programmable architectures that are complex and self-organized: from morphogenesis to engineering', *11th International Conference on the Simulation and Synthesis of Living Systems*, Winchester, UK, August 5–8, ALIFE XI.

Doyle, J. and Csete, M. (2007) 'Rules of engagement', *Nature*, Vol. 446, No. 7138, p.860.

Dressler, F. (2007) *Self-Organization in Sensor and Actor Networks*. NY: Wiley.

Dunn, M. and Mauer, V. (2006) (Eds.) *The International Critical Information Infrastructure Protection (CIIP) Handbook 2006: Analyzing Issues, Challenges, and Prospects*. ETH Zürich: Center for Security Studies.

EE (2002) *Workshop on Emergent Engineering*. Boston, MA: MIT. Available at: http://cba.mit.edu/events/02.10.emergent (accessed 6 October 2002).

Gierer, A. and Meinhardt, H. (1972) 'A theory of biological pattern formation', *Kybernetik*, Vol. 12, pp.30–39.

Grégoire, G. and Chaté, H. (2004) 'Onset of collective and cohesive motion', *Physical Review Letters*, Vol. 92, p.025702.

Grobbelaar, S. and Ulieru, M. (2007) 'Complex networks as control paradigm for complex systems', *1st International Workshop on eNetworks Cyberengineering, IEEE Systems Man and Cybernetics Conference*, Montreal, Canada, October 7–10.

Gross, R., Bonani, M., Mondada, F. and Dorigo, M. (2006) 'Autonomous self-assembly in swarm-bots', *IEEE Transactions on Robotics*, Vol. 22, No. 6, pp.1115–1130.

Holland, J. (1998) *Emergence: From Chaos to Order.* Redwood City, CA: Addison-Wesley.

Isermann, R. (1996) *Digital Control Systems*. New York: Springer-Verlag.

IST (2006) *Workshop on Resilient Infrastructures and Information Fusion for Security*, EU IST Event 2006, Helsinki, Finland, November 21–23.

IT Revolutions (2008) 'Foreword by Mihaela Ulieru, General Chair', *1st International Forum of an IT-Driven World*, Venice, Italy, December 17–19.

ITU Internet Reports (2005) 'The internet of things', *World Summit of the Information Society*, Tunis, Tunisia, November 16–18. Available at: www.itu.int/internetofthings

Kauffman, S. (2000) *Investigations*. Oxford University Press.

Kauffman, S. (2008) *Reinventing the Sacred*. Basic Books.

Kicinger, R. (2004) *Emergent Engineering Design: Design Creativity and Optimality Inspired by Nature*, PhD Thesis, George Mason University, 680 pages; AAT 3151149.

Lee, E.A. (2007) 'Computing foundations and practice for cyber-physical systems: a preliminary report', *Technical Report UCB/EECS-2007-72*, University of California, Berkeley.

Levin, S.A. (2003) 'Complex adaptive systems: exploring the known, the unknown and the unknowable', *Bulletin of the American Mathematical Society*, Vol. 40, pp.3–19.

Macal, C.M. and North, M.J. (2006) 'Tutorial on agent-based modeling and simulation, Part 2: how to model with agents', in L.F. Perrone et al. (Eds.), *Proceedings of the 2006 Winter Simulation Conference*, pp.73–83.

Marzano, S. and Aarts, E. (2003) *The New Everyday View on Ambient Intelligence.* The Netherlands: Uitgeverij 010 Publishers.

Minai, A.A., Braha, D. and Bar-Yam, Y. (2006) 'Complex engineered systems', in D. Braha, Y. Bar-Yam and A.A. Minai (Eds.), *Complex Engineered Systems: Science Meets Technology*. Springer-Verlag.

Müller-Schloer, C. and Sick, B. (2008) 'Controlled emergence and self-organization', in R.P. Würtz (Ed.), *Organic Computing*. Springer-Verlag, pp.81–104.

Nagpal, R. (2002) 'Programmable self-assembly using biologically-inspired multi-agent control', *1st International Conference on Autonomous Agents*, Bologna, Italy, July 15–19.

Newman, M.E.J. (2006) 'Modularity and community structure in networks', *Proceedings of the National Academy of Sciences*, Vol. 103, No. 23, pp.8577–8582.

North, M.J. and Macal, C.M. (2007) *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford University Press.

SCADA (2006) *Proceedings of the 2006 Process Control and SCADA Security Summit*, Las Vegas, Nevada, September 28–30.

Siero, P., Rozenberg, G. and Lindenmayer, A. (1982) 'Cell division patterns: syntactical description and implementation', *Computer Graphics and Image Processing*, Vol. 18, pp.329–346.

Stanley, K.O. and Miikkulainen, R.A. (2003) 'Taxonomy for artificial embryogeny', *Artificial Life*, Vol. 9, pp.93–130.

Tanenbaum, A.S. and van Steen, M. (2002) *Distributed Systems: Principles and Paradigms*. Prentice Hall.

Terranova, T. (2004) *Network Culture: Politics for the Information Age*. Pluto Press.

Ulieru, M. (2004) 'Emerging computing for the industry: agents, self-organization and holonic systems', *Workshop on Industrial Informatics*, Busan, South Korea, November 2–6, IECON'04.

Ulieru, M. (2008) 'Enabling the SOS network', *Proceedings of the IEEE Systems, Man and Cybernetics Conference* (SMC 2008), Singapore, October 12–15.

Willinger, W. and Doyle, J. (2002) 'Robustness and the internet. Design and evolution', in E. Jen (Ed.), *Robust Design: A Repertoire of Biological, Ecological, and Engineering Case Studies*. Oxford University Press, pp.231–272.

Wooldridge, M. (2002) *An Introduction to Multiagent Systems*. John Wiley and Sons Ltd.

Würtz, R.P. (2008) (Ed.) *Organic Computing*. Understanding Complex Systems. Springer-Verlag.

Notes

[1]For previous works of the authors, on which this paper builds further, please refer to their websites.

[2]Complexity here is regarded as a *collective behaviour* resulting from interaction between parts, which cannot be anticipated because it is not implicitly contained in the behaviour of the individual parts at a particular scale of observation. Emerging properties of the collective behaviour are novel with respect to the individual parts of the system (Holland, 1998).