Chapter 15
The Computing of Digital Ecosystems

Gerard Briscoe
London School of Economics and Political Science, UK

Philippe De Wilde
Heriot-Watt University, UK

ABSTRACT

A primary motivation this research in digital ecosystems is the desire to exploit the self-organising properties of biological ecosystems. Ecosystems are thought to be robust, scalable architectures that can automatically solve complex and dynamic problems. However, the computing technologies that contribute to these properties have not been made explicit in digital ecosystems research. In this paper, the authors discuss how different computing technologies can contribute to providing the necessary self-organising features, including Multi-Agent Systems (MASs), Service-Oriented Architectures (SOAs), and distributed evolutionary computing (DEC). The potential for exploiting these properties in digital ecosystems is considered, suggesting how several key features of biological ecosystems can be exploited in Digital Ecosystems, and discussing how mimicking these features may assist in developing robust, scalable self-organising architectures. An example architecture, the Digital Ecosystem, is considered in detail. The Digital Ecosystem is then measured experimentally through simulations, which consider the self-organised diversity of its evolving agent populations relative to the user request behaviour.

INTRODUCTION

Digital Ecosystems are distributed adaptive open socio-technical systems, with properties of self-organisation, scalability and sustainability, inspired by natural ecosystems (Briscoe, 2009), and are emerging as a novel approach to the catalysis of sustainable regional development driven by Small and Medium sized Enterprises (SMEs). Digital Ecosystems aim to help local economic actors become active players in globalisation, valorising their local culture and vocations, and enabling them to interact and create value networks at the global level (Dini et al., 2008).
We have previously considered the biological inspiration for the technical component of Digital Ecosystems (Briscoe, Sadedin, & Paperin, 2007; Briscoe & Sadedin, 2007), and we will now consider the relevant computing technologies. Based on our understanding of biological ecosystems in the context of Digital Ecosystems (Briscoe & De Wilde, 2006), we will now introduce fields from the domain of computer science relevant to the creation of Digital Ecosystems. As we are interested in the digital counterparts for the behaviour and constructs of biological ecosystems, instead of simulating or emulating such behaviour or constructs, we will consider what parallels can be drawn. We will start by considering MASs to explore the references to agents and migration (Briscoe et al., 2007; Briscoe & Sadedin, 2007); followed by evolutionary computing and SOAs for the references to evolution and self-organisation (Briscoe et al., 2007; Briscoe & Sadedin, 2007).

The value of creating parallels between biological and computer systems varies substantially depending on the behaviours or constructs being compared, and sometimes cannot be done so convincingly. For example, both have mechanisms to ensure data integrity. In computer systems, that integrity is absolute, data replication which introduces even the most minor change is considered to have failed, and is supported by mechanisms such as the Message-Digest algorithm 5 (Rivest, 1992). While in biological systems, the genetic code is transcribed with a remarkable degree of fidelity; there is, approximately, only one unforced error per one hundred bases copied (McCulloch et al., 2004). There are also elaborate proof-reading and correction systems, which in evolutionary terms are highly conserved (McCulloch et al., 2004). In this example establishing a parallel is infeasible, despite the relative similarity in function, because the operational control mechanisms in biological and computing systems are radically different, as are the aims and purposes. This is a reminder that considerable finesse is required when determining parallels, or when using existing ones.

**MULTI-AGENT SYSTEMS**

A software agent is a piece of software that acts, for a user in a relationship of agency, autonomously in an environment to meet its designed objectives. A MAS is a system composed of several software agents, collectively capable of reaching goals that are difficult to achieve by an individual agent or monolithic system. Conceptually, there is a strong parallel between the software agents of a MAS and the agent-based models of a biological ecosystem (Green, Klomp, Rimmington, & Sadedin, 2006), despite the lack of evolution and migration in a MAS. There is an even stronger parallel to a variant of MASs, called mobile agent systems, in which the mobility also mirrors the migration in biological ecosystems (Pham & Karmouch, 1998).

The term mobile agent contains two separate and distinct concepts: mobility and agency (Rothermel & Hohl, 1999). Hence, mobile agents are software agents capable of movement within a network (Pham & Karmouch, 1998). The mobile agent paradigm proposes to treat a network as multiple agent-friendly environments and the agents as programmatic entities that move from place to place, performing tasks for users. So, on each host they visit mobile agents need software which is responsible for their execution, providing a safe execution environment (Pham & Karmouch, 1998).

Generally, there are three types of design for mobile agent systems (Pham & Karmouch, 1998): (1) using a specialised operating system, (2) as operating system services or extensions, or (3) as application software. The third approach builds mobile agent systems as specialised application software that runs on top of an operating system, to provide for the mobile agent functionality, with such software being called an agent station (McCabe & Clark, 1994). In this last approach, each agent station hides the vendor-specific aspects of its host platform, and offers standardised services to visiting agents. Services include access to local resources and applications; for example, web
Related Content

Optimum Design of Hybrid EDFA/FRA by Particle Swarm Optimization
[www.igi-global.com/chapter/optimum-design-hybrid-edfa-fra/72827?camid=4v1a](www.igi-global.com/chapter/optimum-design-hybrid-edfa-fra/72827?camid=4v1a)

[www.igi-global.com/article/sc-ws/114647?camid=4v1a](www.igi-global.com/article/sc-ws/114647?camid=4v1a)

Compensation of Voltage Sags with Phase-Jumps through DVR with Minimum VA Rating Using PSO based ANFIS Controller
Anil Kumar Ramakuru, Siva G. Kumar, Kalyan B. Kumar and Mahesh K. Mishra (2012). *Innovations and Developments of Swarm Intelligence Applications* (pp. 133-147).
[www.igi-global.com/chapter/compensation-voltage-sags-phase-jumps/65810?camid=4v1a](www.igi-global.com/chapter/compensation-voltage-sags-phase-jumps/65810?camid=4v1a)

Constraint Handling in Particle Swarm Optimization
[www.igi-global.com/chapter/constraint-handling-particle-swarm-optimization/65806?camid=4v1a](www.igi-global.com/chapter/constraint-handling-particle-swarm-optimization/65806?camid=4v1a)