Digital Ecosystems: Evolving Service-Orientated Architectures

Gerard Briscoe
Intelligent Systems and Networks Group
Department of Electrical and Electronic Engineering
Imperial College London
London, United Kingdom, SW7 2BT
e-mail: gerard.briscoe@ic.ac.uk

Philippe De Wilde
Intelligent Systems Lab
School of Mathematical and Computer Science
Heriot-Watt University
Edinburgh, United Kingdom, EH14 4AS
e-mail: pdw@macs.hw.ac.uk

Abstract—A novel optimisation technique inspired by natural ecosystems is presented, where the optimisation works at two levels: a first optimisation, migration of services which are distributed in a decentralised peer-to-peer network, operating continuously in time; this process feeds a second optimisation based on evolutionary computing that operates locally on single peers and is aimed at finding solutions to satisfy locally relevant constraints. Through this twofold process, the local search is accelerated and will yield better local optima, because the distributed optimisation already provides a good sampling of the search space by making use of computations already performed in other peers with similar constraints. We call this new distributed optimisation architecture a Digital Ecosystem, an Ecosystem-Orientated Architecture (EOA) created by extending a Service-Oriented Architecture (SOA) with Distributed Evolutionary Computing (DEC). The Digital Ecosystem will allow services to recombine and evolve over time, constantly seeking to improve their effectiveness for the user base. Individuals within our Digital Ecosystem will be applications (groups of services), created in response to user requests by using evolutionary optimisation to aggregate the services. These individuals will migrate through the Digital Ecosystem and adapt to find niches where they are useful in fulfilling other user requests.

With the ever increasing number of services offered online from Application Programming Interfaces (APIs) being made public, there is an ever increasing number of computational units available to be combined to create applications. However, this is currently a task done manually by programmers. There are several existing efforts aimed at achieving automatic service composition [2], [3], [4], [5], the most prevalent of which is SOAs and its associated standards and technologies [6]. We propose a biologically inspired approach, expected to more affective than traditionally inspired approaches at greater scales, because it is built upon the scalable properties of natural ecosystems. An Ecosystem-Orientated Architecture, called a Digital Ecosystem, which itself is built upon a SOA and that uses a novel form of DEC for the automatic combining of available and applicable services to meet user requests for applications in a scalable architecture.

This work is supported by the European Commission under the EU project Digital Business Ecosystems (contract number 507953 [1]).

I. SERVICE-ORIENTATED ARCHITECTURES

The SOA concept expresses a software architectural concept that incorporates modular reusable services that have clearly defined and standardised interfaces.

In a SOA environment, nodes on a network make resources available to other participants in the network, as independent services that the participants can access in a standardised way. Most definitions of SOAs identify the use of web services in their implementation. Unlike traditional object-oriented architectures, SOAs comprise loosely joined, highly interoperable application services, and because these services interoperate over different development technologies, the software components become very reusable [7].

SOAs potentially promise to provide a huge number of services that programmers can combine, via the standardised interfaces, to create increasingly more sophisticated and distributed applications.

There are multiple standards available, and still being developed, for SOAs [6]. So, we will define how we view services in generic terms, and then we will consider an example implementation. We consider each service to consist of an executable component, available remotely or locally, and a descriptive component. The descriptive component acting as a guarantee of behaviour. The actual underlying code or service only comes into play once the whole application has been assembled. A service can be a software service, e.g. for data encryption, or a real world service, e.g. selling books, which can be represented by a software service. This is widely applicable and easily adaptable definition, and we have used it to consider examples in the travel and manufacturing industries [8].

The Semantics of Business Vocabulary and Business Rules (SBVR) is a near natural language for describing business services. It features strong support for multilingualism, the use of formal logic and has compliance with Meta Object Facility (MOF) [9]. It allows business vocabularies construction, and business rules definitions, enabling their interchange among organisations. Since it allows talking about semantics, vocabulary and rules, this foundation is named as 'Business Vocabulary+Rules'. The SBVR Vocabulary defined
is extensible since it is a vocabulary. It can also be included in other vocabularies to create an extended SBVR Vocabulary. The following is a few vocabulary entries from the SBVR metamodel:

```
car rental
  General Concept: business entity
  Reference Scheme: the name of the car rental

car rental has name

car storage capacity
  Definition: number of cars that can be stored at the car rental site

car rental has car storage capacity
```

The key point to stress is that SBVR descriptions consist of descriptive elements defined by units and subunits, and also data elements when the model is populated. Show below is an SBVR vocabulary for a car rental service:

```
car rental
  General Concept: business entity
  Reference Scheme: the name of the car rental

car rental has name

car storage capacity
  Definition: number of cars that can be stored at the car rental site

car rental has car storage capacity
```

The following statement is an SBVR service description:

```
The car rental ‘Hertz’, has cars storage capacity 200.
```

II. DISTRIBUTED EVOLUTIONARY COMPUTING

Evolutionary Computing (EC) is an optimisation technique capable of finding solutions in a large search space of possible solutions. It works on the principle of using evolution to generate a solution to a specific problem. The cyclic process of mutation, replication and death, is applied to a population of possible solutions. The selection pressure comes from a fitness function which is instantiated with the specific problem for which a solution is to be found. The fitness function determines the fitness of the different solutions relative to the environment, and therefore their replication rate. So the better solutions get to replicate more, potentially providing improved solutions in the next generation.

DEC aims to achieve parallel processing to find solutions faster. We will focus on the Island Model which is probably the most analogous to the notion of migration in nature [10]. There is a distance between the sub-populations on each island, and a probability of migration between one island and another. This model has been used successfully in the determination of investment strategies in the commercial sector, in a product known as the Galapagos toolkit [11].

The Island Model is visualised in Figure 1, in which there are different probabilities of going from island ‘1’ to island ‘2’ as there is of going from island ‘2’ to island ‘1’. This allows maximum flexibility for the migration process. It also mirrors the naturally inspired quality that, although two populations have the same separation no matter how you look at them, it may be easier for one population to migrate to another than vice-versa. An example of this from nature could be the situation whereby it is easier for one species of fish to migrate down-stream than for a different species to migrate upstream. However, all the islands in this approach work on exactly the same problem, which makes it less analogous to biological ecosystems in which different locations can be environmentally different. We will take advantage of this later in the EOA we create.

III. THE DIGITAL ECOSYSTEM

The EOA of our Digital Ecosystem represents a novel approach to DEC inspired from nature’s ecosystems. An ecosystem is a biological community (a set of organisms from different species interacting together) in conjunction with its physical environment, and a part of this physical environment where a particular species lives is called a habitat [12]. Each user interacts with a node to access the services of the
SOA, and we have enhanced this node to operate equivalently to a biological habitat, being a more sophisticated version of an island from the Island Model of DEC. We call this enhanced node a habitat, with a network of habitats creating the Digital Ecosystem (Figure 2). Therefore, the organisms of the Digital Ecosystem are the services of the SOA offered on the network. The habitats are connected through a peer-to-peer decentralised network. The connections between peers are dynamically adapted on the basis of observed migration paths of the services used between users within the habitat network. Following the idea of Hebbian learning, the habitats which exchange individuals more often obtain stronger connections and habitats which do not exchange individuals will become less strongly connected. This way, a network topology is discovered with time, which reflects the structure of the business-sectors within the user base. The resulting network will resemble the connectivity of the businesses within the user base, typically a small-world network in the case of Small and Medium sized Enterprises (SMEs) [13], [14], [15]. We have focused on SMEs since they make up 99% of enterprises in most EU Member States [16]. Such a network has many strongly connected clusters called sub-networks (quasi complete graphs) and a few connections between these clusters. Graphs with this topology have a very high clustering coefficient and small characteristic path lengths [17], [18], as shown in Figure 3.

In the Digital Ecosystem, local and global optimisations operate concurrently in finding solutions to satisfy different optimisation problems. The global optimisation here is not a super-peer based central control mechanism, but a completely distributed one which avoids a single point of failure. This allows for an optimised dynamic orchestration of services and their combinations, as well as a high degree of learning capabilities for the services and for the infrastructure itself.

A. Services

An organism in the Digital Ecosystem is actually a pointer to the software service it represents, which leads to the service’s semantic description and executable component, whether they are local or remote. So, from a logical point of view the individuals are equivalent to the services they represent, and we will therefore refer to the services represented instead of the pointers that represent them.

The service’s semantic description acts as a guarantee of its functionality, and is the inheritable component from one generation to the next in the evolutionary optimisation. In an evolving population, the fitness of a combination of services (application) will be based on comparing the descriptive components of the services with the complex description of the user request, complex in the sense that more than one service is required to fulfil the request.

The service is the base unit for the evolutionary mechanism, which means that the evolutionary process optimises the combination of services to a user request. It does not change the services themselves. The aggregated structure of services is a set. Therefore, the service-pool of a habitat is the set of service(-set)s available at a habitat, and it will vary between the set of services available at the habitat and the power set of the available services.

B. Evolving Populations

An evolving population, similar to the evolutionary aspects of an island from the Island Model of DEC, will be created to use EC and evolve the fittest (optimal) solution(s) to a request defined by the user. The population is seeded with the contents of the service-pool from the habitat in which the evolving population is instantiated. This allows the evolutionary optimisation to be bootstrapped by previous solutions stored in the service-pool. The evolving population of services will then search the available service combination space through evolution to find the optimal solution(s) to a user request. The fitness of individual service-sets within a population will be determined by a selection pressure applied as a fitness function instantiated from the user request, and works primarily on comparing the semantic descriptions of the services with the semantic description in the user request. Mutations can occur by switching services in and out of the service-set structure. Recombination (Crossover) can occur by combining elements of two service-sets into a new service-set. The optimal solution (service-set) found can then be migrated through the interconnected habitats, recombining with other services to meet more user requests in other Evolving Populations.

C. Habitats

The habitat is the key component for supporting the ecosystem concept within the EOA. The Digital Ecosystem will consist of interconnected habitats just as in a biological ecosystem. Each user has a dedicated habitat node, which includes a service-pool of service(-set)s that are of potential use to the user, and which acts as a gene pool for the evolving populations instantiated within the habitat. New solutions evolved from the service-pool in response to user requests are also stored within the service-pool, as the combination of services has value as well as the services themselves. The habitat also provides a place for service migration to occur, as the habitats are interconnected with one another. The union of the habitats creates the EOA of the Digital Ecosystem.

The novelty of our approach comes from the evolving populations being created in response to similar requests. So
whereas in the Island Model (Figure 1) there are multiple evolving populations in response to one request, in the Digital Ecosystem there will be multiple evolving populations in response to similar requests. This is shown in Figure 4 where the shades of the evolving populations indicates similarity in the requests being managed. For example, the four habitats in the top left of Figure 4 may all be travel agents, the three with evolving populations may be looking for package holidays within the same continent. A great holiday package (service-set) found and used in one habitat, will then be migrated to the other connected habitats where it will be integrated into any currently active evolving populations via the local service-pool. So this will potentially help to optimise the search of similar package holidays at the habitats of the other travel agents. This will also work in a time-shifted manner, because the great holiday package (solution) will be stored in the service-pools of the habitats to which it is migrated. So it will potentially be available to optimise similar requests placed later.

Our approach of utilising similar requests does require that the Digital Ecosystem has sufficient critical mass in the user base, for there to be similarity in the requests of users within the community. Assuming there is hundreds of habitats there will be potentially three or more times the number of Evolving Populations at any one time. There will then be thousands of services, and a potentially huge number of possible service-set combinations (applications). In such a scenario, there would be a sufficient number of user requests for the Digital Ecosystem to find similarity within them to apply the novel DEC approach of the EOA.

D. Service Migration

The inter-habitat connectivity is the migration of services from one habitat to another. Each connection between the habitats is bi-directional, and there is a probability associated with moving in either direction across the connection (like the connections of the DEC Island Model shown in Figure 1). The connection probabilities affect the migration of services and are updated by the success or failure of the migration.

The migration of a service is triggered by it being deployed into its Home habitat or when it is used by a user at any habitat. When migration occurs, depending on the probabilities associated with the connections; an exact copy of the service is made at the connected habitat. The copy of the service is identical, then its migration history is updated which will differentiate it from the original.

The success of the migration, the migration feedback, is determined by the usage of services at the habitats to which they migrate. When a solution (service-set) is found to a user request, then the individual service migration histories can be used to determine where the used service has come from and update the connection probabilities. The challenge is in managing the feedback to the connection probabilities for migrating service-sets, because for a service-set the value is within the combination. The benefit of where the combination or subsets of the combination were created needs to be passed on to the connection probabilities. Specifically, the mechanism needs to know where used service-sets are created. So that, new connections can be created or existing connections can be reinforced, to where the service-sets where created.

The escape range is the number of escape migrations possible upon the risk of death. If a service migrates to a habitat and remains unused after several user requests, then it will have the opportunity to migrate (moved not copied) randomly to a another connected habitat. After this happens several times the service will be deleted (die). The escape range is a parameter that will be better determined by future work and practical experience. Ideally the escape range will be adjusted to the size of the habitat cluster it exists within. This is not obvious, because the view of a cluster very much depends on the point of view of the habitat. Creating a definition that is dynamically responsive to the state of the system is the preferred solution. This would effectively create a dynamic time-to-live for the services, but even so, services that are used will live longer than those that are not.

E. Habitat Clustering

The connectivity in the habitats will be parallel to the existing communities that are within the user base, but the infrastructure also allows for the spontaneous formation of new communities via clustering in the habitat network. As shown in Figure 5, many successful service migrations will reinforce habitat connections, thereby increasing the probability of service migration. If a successful migration occurs through multi-hop migration, then a new link can be formed between those habitats. Unsuccessful migrations will lead to connections (migration probabilities) decreasing, until finally the connection is closed.

Each cluster of habitats represents a community within the user base. The clustering/community formation is like joining a club; you can be a member of more than one. In the same way habitats can be clustered into more than one
community. The clustering will lead to communities clustered over language, business sector, nationality, geography, etc.

IV. SIMULATION AND RESULTS

The Digital Ecosystem was simulated to test the efficiency of the system in responding to user requests. Throughout the simulation we assumed a hundred users, which meant that at any time, the number of users that joined the network was equal to the number of users that would leave. The users’ habitats were initially randomly connected. At the start, users deployed five services each. Services could migrate from one habitat to another thereby affecting the connections between the habitats. Users would submit requests for services. Each user then deployed a new service after the submission of three requests. The order in which users submitted requests was random.

An important measure for determining the success of the Digital Ecosystem is its relative performance to a traditional SOA based system. We simulated a simple SOA with a distributed service registry. A typical simulation run of the Digital Ecosystem under the above conditions was compared with a typical run of the SOA based system. The time available to the SOA based system for the search in the distributed registry was limited to the time the Digital Ecosystem required to perform the same user request. In Figure 6 we graphed the percentage match to the user requests for typical runs, as determined by a distance function between the request and the service descriptions.

Both the Digital Ecosystem and the SOA based system performed as expected, providing better responses to user requests as more services became available. The SOA reference system initially performed better than the Digital Ecosystem, but with the increasing number of services the Digital Ecosystem began to outperform the reference system and continued to do so. This was anticipated as the Digital Ecosystem was expected to be more effective at larger scales [19].

V. CONCLUSION

A hybrid platform combining SOAs with DEC has been created that is the digital counterpart of an ecosystem. We have created a complex system to provide a scalable solution that is in essence greater than the sum of its parts, and expected to show emergent and complex behaviour that cannot be predicted until it is created. The experimental results indicate that under simulation conditions the Digital Ecosystem outperforms the comparison system based on a traditional SOA.

The Digital Ecosystem model was presented at the 2006 Java One conference [19], and a dedicated open-source simulation framework has been created to assist the ongoing investigations and research into the Digital Ecosystem [20].

A SOA to which this approach is being applied is the Digital Business Ecosystem (DBE), an EU Framework VI project [1]. The DBE aims to be a next generation information and communications technology. In essence, the DBE will be an Internet-based environment in which businesses will be able to interact with each other in very effective and efficient ways. We have created a prototype Digital Ecosystem within the DBE, called the Evolutionary Environment (EvE), constructed upon the SOA of the DBE, the Execution Environment (ExE) [21], [22], which will use the Evolutionary Computing in Java (ECJ) library for the evolutionary optimisation. The core of the EvE Digital Ecosystem is currently being implemented [23] in the DBE project by Salzburg University of Applied Sciences, Sun Microsystems Iberica, TechIDEAS, and Intel Ireland. Soon it will be possible to collect real world data to investigate further the complex nature of a Digital Ecosystem.

ACKNOWLEDGMENT

The authors would like to thank for their encouragement and suggestions: Paolo Dini of the London School of Economics and Political Science, Thomas Heistracher and his group of the Salzburg University of Applied Sciences, Jonathan Rowe of the University of Birmingham, and Miguel Vidal of
Sun Microsystems Iberica. The Digital Ecosystem model was constructed through interacting with these people and others.

REFERENCES

[1] E. L. Project, “Digital business ecosystems project.” [Online]. Available: http://www.digital-ecosystem.org/

[2] N. Milanovic and M. Malek, “Current solutions for web service composition,” Internet Computing, IEEE, vol. 8, no. 6, pp. 51–59, 2004.

[3] M. Bossardt, R. Antink, A. Moser, and B. Plattner, Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2004, vol. 2982, ch. Chameleon: Realizing Automatic Service Composition for Extensible Active Routers, pp. 163–177.

[4] S. McIlraith, C. Son, and H. Zeng, “Semantic web services,” Intelligent Systems, IEEE, vol. 16, no. 2, pp. 46–53, 2001.

[5] J. Rao and X. Su, Lecture Notes in Computer Science. Berlin, Heidelberg: Springer-Verlag, 2005, vol. 3387, ch. A Survey of Automated Web Service Composition Methods, pp. 43–54.

[6] D. Norfolk, “Avoiding soa standards-based chaos,” in Service-Oriented Architecture, Concepts, Technology and Design. Prentice Hall PTR, 2005.

[7] G. Briscoe and P. D. Wilde, “D6.6 high-level design specification of the distributed intelligence system,” Digital Business Ecosystem, vol. Contract no 507953, 2006.

[8] M. D. Tommasi, “D15.3 BML framework 2nd release,” Digital Business Ecosystem, 2005.

[9] L. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.

[10] M. Ward, “Life offers lessons for business.” [Online]. Available: http://news.bbc.co.uk/2/hi/technology/3752752.stm

[11] M. Begon, J. Harper, and C. Townsend, Ecology: Individuals, Populations and Communities. Third Edition. Blackwell Publishing, 1996.

[12] X. Yang, “Chaos in small-world networks,” PHYSICAL REVIEW E, vol. 63, no. 046206, 2001.

[13] Censis, “Deliverable 27.1: Territorial social capital and driver smes,” Digital Business Ecosystem Projects Report.

[14] O. of Europeans SMEs, “Smes and cooperation,” European Commission, 2003.

[15] F. Nachira, “Towards a network of digital business ecosystems: digital business ecosystems fostering the local development,” European Commission DG INFSO. Tech. Rep., 2002.

[16] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” Nature, vol. 393, no. 6684, pp. 440–442, 1998.

[17] M. E. J. Newman and D. J. Watts, “Scaling and percolation in the small-world network model,” Physical Review E 60, vol. 6, pp. 7332–7342, 1999.

[18] G. Briscoe, M. Chli, and M. Vidal, “BOF-0759 Creating a Digital Ecosystem: Service Oriented Architectures with Distributed Evolutionary Computing,” in JavaOne Conference, 2006. [Online]. Available: https://www28.cpan.com/javaone06/\_cv\_124\_1/session\_details.jsp?isid=277759&\_location\_id=124-1&\_language=english

[19] T. Kurz. [Online]. Available: http://eveim.sourceforge.net/

[20] Digital Business Ecosystem Project, “Execution Environment.” [Online]. Available: http://swallow.sourceforge.net/

[21] T. Heisteracher, T. Kurz, C. Masuch, P. Ferronato, M. Vidal, A. Corallo, P. Dini, and G. Briscoe, “Pervasive service architecture for a digital business ecosystem,” in ECOOP First International Workshop on Coordination and Adaptation Techniques for Software Entities (WCAT04), June 2004, http://wcat04.unex.es/.

[22] C. Masuch, “Evolutionary environment network.” [Online]. Available: http://eveim.sourceforge.net/

[23] B. Sleeper and B. Robins, “Defining Web Services.” [Online]. Available: http://www.perfectxml.com/Xanalysis/TSG/TSG\_DefiningWebServicespdf

[24] W. G. Hale, J. P. Margham, and V. A. Saunders, Dictionary of BIOLOGY, 2nd ed. HarperCollins Publishers Inc., 1995.

[25] A. Redmore and M. Griffin, Longman Reference Guides: Biology, 7th ed. Longman Group Limited, 1994.

[26] W. Beck, K. Liem, and G. Simpson, LIFE: An Introduction to Biology, 3rd ed. HarperCollins Publishers Inc., 1991.

[27] W3C, “Web Services Activity” 2004. [Online]. Available: http://www.w3.org/2002/ws/

ABOUT THE AUTHORS

Gerard Briscoe is a Research Assistant at the Intelligent Systems Lab of the School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, United Kingdom, while working towards his Ph.D. at the Intelligent Systems and Networks group of the Department of Electrical and Electronic Engineering, Imperial College London, United Kingdom. He is also a visiting academic at the Digital Ecosystems Lab of the Department of Media and Communications at the London School of Economics and Political Science, United Kingdom. His research interests focus on computational biomimicry, which is the application of models, theories and ideas from biology to software systems. This includes evolutionary computing and Biological Design Patterns (BDPs). He received his Masters of Engineering degree from the Department of Computing, Imperial College London, in 2002. He is a member of the Association for Computing Machinery.

Philippe De Wilde is a Professor at the Intelligent Systems Lab, School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, United Kingdom. Research interests: stability, scalability and evolution of multi-agent systems; networked populations; coordination mechanisms for populations; group decision making under uncertainty; neural networks, neuro-economics. He tries to discover biological and sociological principles that can improve the design of decision making and of networks. Research Fellow, British Telecom, 1994. Laureate, Royal Academy of Sciences, Letters and Fine Arts of Belgium, 1988. Senior Member of IEEE, Member of IEEE Computational Intelligence Society and Systems, Man and Cybernetics Society, ACM, and British Computer Society. Associate Editor, IEEE Transactions on Systems, Man, and Cybernetics, Part B, Cybernetics.