Dependable Composition of Software and Services in the Internet of Things: A Biological Approach

Amleto Di Salle, Francesco Gallo(✉), and Alexander Perucci

University of L'Aquila, Via Vetoio 1, 67100 L'Aquila, Italy
{amleto.disalle,francesco.gallo}@univaq.it,
alexander.perucci@graduate.univaq.it
http://www.univaq.it

Abstract. If we pause a moment to reflect on the innovations of the last twenty years in the field of information technology, we realize immediately as consumer electronics, computers and telecommunications have changed their balance of power. Today, Internet is so ingrained in the culture of the people who seems to be always there, and you can hardly imagine to live without it. Today mobile devices are computers, and their inter-connection and connection with several kinds of technological objects is ever more increasing. This has led to the emergence of new concepts, such as the Internet of Things (IoT), Machine to Machine (M2M), and People to Machine (P2M) and the consequent need to provide frameworks that allows communication and interoperability between heterogeneous objects. Furthermore, the increasing availability of data and especially of computational power allows *things* to serve not only as data producers but also as consumers.

Thus we can think about *internet of things as a cloud of services* software and services composition.

In such a highly mobile environment, both the user and "things" may be subject to frequent movement, demanding a frequent recomposition. In this paper, we propose a preliminary biological-inspired approach for adaptive software composition at run time. The approach leverages the concept of immune system to ensure dependability e.g. availability and reliability, of a composition of software and services in the Internet of Things.

Keywords: Adaptive composition · Run-time composition · Service composition

1 Introduction

Information and communication systems are invisibly embedded in the environment around us. This results in the generation of enormous amounts of data which have to be stored, processed and presented in a seamless, efficient and easily interpretable form. This model will consist of services that are commodities and delivered in a manner similar to traditional commodities.

D. Bianculli et al. (Eds.): SEFM 2015 Workshops, LNCS 9509, pp. 312–323, 2015.
DOI: 10.1007/978-3-662-49224-6_25

The computing criterion will need to go beyond traditional mobile computing scenarios that use smart phones and portables, and evolve into connecting everyday existing objects and embedding intelligence into our environment.

The traditional Internet concept is radically evolving into a Network of interconnected objects that not only harvest information from the environment (sensing) and interacts with the physical world (actuation/command/control), but also uses existing internet standards to provide services for information transfer, analytics, applications and communications. So, we have seen the birth of new concepts such as *Internet of Things*, that [1,2], define as: (1) *'Things' are active participants in business, information and social process where they are enabled to interact and communicate among themselves and with the environment by exchanging data and information sensed about the environment, while reacting autonomously to real/physical world events and influencing it by running processes that trigger actions and create services with or without direct human intervention*, and (2) *Interconnection of sensing and actuating devices providing the ability to share information across platforms through a unified framework, developing a common operating picture for enabling innovative applications. This is achieved by seamless large scale sensing, data analytics and information representation using cutting edge ubiquitous sensing and cloud computing.*

These definitions promote the development of novel tools for the agile, transparent, and automated composition of objects in heterogeneous environments, dominated by a high dynamism. Dynamism involves access to resources and services that can not be guaranteed over time, causing the user to perceive the "Net" as not dependable. Being inspired by biological systems, an immune system is a form of highly dynamic system by its nature. In particular, an immune system is composed of loosely-coupled elements, which can interact with each other by exchanging asynchronous messages, providing a high capability to composition. Moreover, it can react to unforeseen events, e.g., intrusions, and it has the ability to adapt to context changes and provide the correct behaviour, making the system dependable and responsive.

In this paper, we propose a biological-inspired preliminary approach that leverages the peculiarities of immune systems so to allow the composition of software and services in a dynamic and reliable way, in order to provide a tool able to manage the **combination** and **selection** of software and services in the context of IoT. This composition is produced in order to satisfy both functional and non functional requirements, such as dependability. Moreover, given the need to write data centric applications (e.g. "Big Data"), of which the Internet of things is large producer, and the need to have applications with the ability to scale horizontally and remaining dependable, however, it has suggested to adopt the use of paradigm functional programming based because, in our view, offers effective techique for the challenges of our context, see Sect. 4.

The paper is organized as follows: in the Sect. 2, we provide an overview about immune systems by also discussing their main elements. In Sect. 3, we introduce basic concepts about software and services composition, by also discussing the new challenges to be faced in the Internet of Things context. In Sect. 4, we present

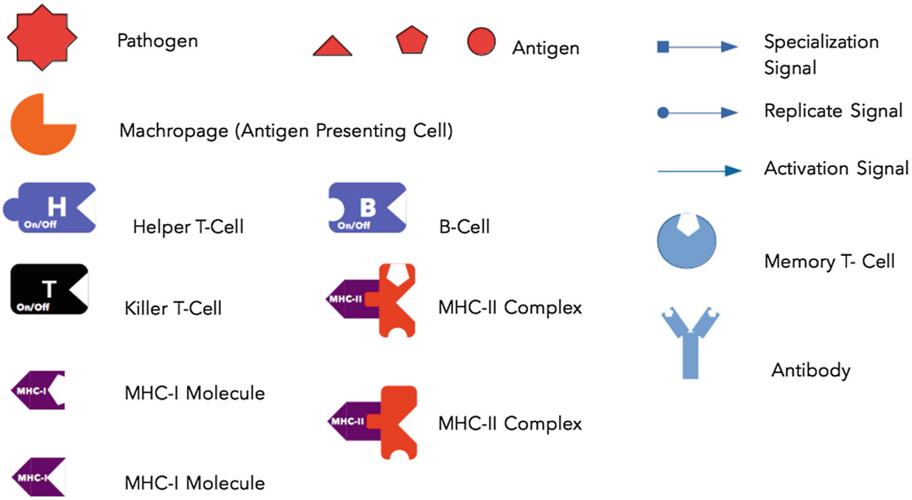


Fig. 1. Immune system components

our biological-inspired approach for the dynamic and dependable composition of software and services in the Internet of Things. We provide a preliminary implementation of the proposed approach that makes use of functional programming. Section 5 concludes and discusses future research directions we should undertake towards the realization of our preliminary proposal.

2 Immune System Aspects

An immune system [3] is a particular kind of biological system that is self-protecting against diseases. It is made of biological structures and processes within an organism. The minimum biological structure within an immune system is the cell, which in turn is made of molecules.

A key feature of an immune system is the ability to distinguish between (i) non-infectious structures, which must be preserved since they do not represent a disease, and (ii) infectious structures, i.e., *pathogens*, which must to be removed since they result in injuries to the organism the immune system belongs to.

But above all, its action immunizing, is performed through the selection and composition of components better suited to deal with the threat.

By referring to [4], the main elements of an immune system can be summarized as follows:

- **Lymphocytes**. They are the cells of an immune system and, for the purposes of this paper, it is enough to distinguish between *T Cells* and *B Cells*:
  - **T Cells**. They can be of two kinds, *T Helper* and *T Killer*. The former are cells responsible for preventing infections by managing and strengthening the immune responses enabled by the recognition of antigens. The

latter are cells able to destroy certain tumor cells, viral-infected cells, and parasites. Furthermore, they are responsible for down-regulating immune responses, when needed.

  - **B Cells**. They are responsible for producing *antibodies* in response to foreign proteins of bacteria, viruses, and tumor cells. B cells are continuously produced in the bone marrow. When the B cell receptor, on the surface of the cell, matches the detected antigens present in the body, the B cell proliferates and secretes a free form of those receptors (antibodies) with identical binding sites as the ones on the original cell surface. The interesting aspect is that the different variants are generated in a seemingly random fashion. Versions that are not biologically significant or useless are automatically deleted.

  Both B Cells and T Cells carry receptor molecules that make them able to recognize specific pathogens. In particular, T Killers recognize pathogens only after antigens have been processed and presented in combination with a *Major Histocompatibility Complex* (MHC) molecule. In contrast, B Cells recognize pathogens without any need for antigen processing.

- **Macrophages**. They are important in the regulation of immune responses. They are often referred to as *Antigen-Presenting Cells* (APC) because they pick up and ingest foreign materials and present these antigens to other cells of the immune system such as T Cells and B Cells. This is one of the important first steps in the initiation of an immune response.
- **Memory Cells**. When B Cell and T Cell are activated and begin to replicate, some cells belonging to their progeny become long-lived *Memory Cells*. The role of Memory Cells is to build an immunological memory that makes the immune system stronger in being self-protecting to future infections/attacks. In particular, a Memory Cell remembers already recognized antigens and lead to a stronger immune response when these antigens are recognized again.
- **Immune response**. An immune response to foreign antigens requires the presence of APC in combination with B Cells or T Cells. When an APC presents an antigen to a B Cell, the B Cell produces antibodies that specifically bind to that antigen in order to kill/destroy it. If the APC presents an antigen to a T Cell, the T Cell becomes active. Active T Cells essentially proliferate and kill target cells that specifically express the antigen presented by the APC. The production of antibodies and the activity of T Killers are highly regulated by T Helpers. They send *signals* to T Killers in order to regulate their activation, proliferation (replication) and efficiency (specialization).

Antibodies are protein complexes with a modular structure which share a basic structure which, in turn, show considerable variability in specific regions capable of binding to structurally complementary particles as antigens. Figure 2 represents the antibodies structure. The variable part, i.e., the V-shaped one, forms the binding site for antigens.

The peculiarity of antibodies, as crucial entities of immune systems, is the specificity with which they are able to recognize antigens. They are able, in fact, to distinguish between protein fragments which differ even for a single amino

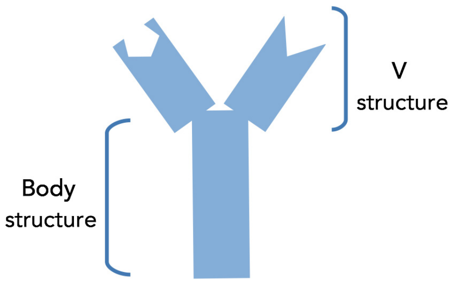


Fig. 2. Antibody example

acid, but also to bind more different antigens. The specificity, however, comes from the ability to create always different antibodies which can bind different antigens capable of building a repertoire of antibodies extremely broad.

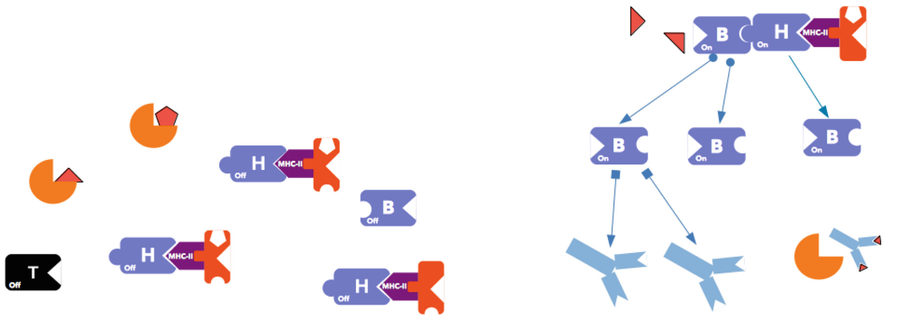
Following the description above, Fig. 1 shows the main elements of an immune system as constructs of a simple graphical notation that we use in Sect. 2.1 for immune system modeling purposes. The figure shows also the messages (signals) that the system elements can exchange.

2.1 Immune System Scenarios

By leveraging the graphical modeling notation introduced above, in this section, we briefly describe two scenarios in the immune systems domain, which are representative for adaptive systems in general. The scenarios provide the reader with a high-level description of the interactions that happen among the elements of an immune system during an immune response.
Scenario 0. Figure 3(a) shows a scenario where the elements of an immune system are in an *inactive* state, marked with the *off* label. In particular, two APC engulf a virus or bacteria: each APC decomposes the pathogen (virus or bacteria) and exposes on its surface a piece of the pathogen, i.e., an antigen (see the triangle and pentagon in the figure). Metaphorically, we can think of this as a setup phase of a computer system, where each system's component is in an idle state and the antigen is a "perturbation" that comes from the outside or even by the system itself. In our context, we can see this perturbation as new system behavior or goal to achieve.
Scenario 1. In this scenario, a B Cell becomes active after that it caught a known antigen and the T Helper close to it became active. In particular, we note that B-cell produces antibodies that exhibit an area of contact (V), complementary to the antigen, which is then captured and subsequently engulfed.

These simple, yet representative scenarios, lead us to observe that, as a particular kind of adaptive and dependable system in a specific domain:

- an immune system is composed of loosely-coupled elements;



(a) Immune system - Scenario 0 (b) Immune system - Scenario 1

Fig. 3. Scenario example

- the elements of an immune system can interact with each other by exchanging asynchronous messages, providing a high capability to composition;
- an immune system can react to unforeseen events, e.g., intrusions;
- the elements of an immune system have the ability to adapt to context changes and provide the correct behaviour, making the system dependable and responsive.

3 Software and Services Composition

In the previous section it was emphasized that the immune response is based on the ability of its components to respond efficiently and effectively to external attacks or malfunctioning, through their composition.

In the traditional context of services composition, an implicit assumption is that services are running on an *enterprise service bus* (ESB) [5]. This proprietary product hide within itself the complexity of the coordination of the different components. Moreover service provider hosts a network accessible software module (an implementation of a given service). The service provider defines a service description of the service and publishes it to a client (or service discovery agency) through which a service description is published and made discoverable. The service client (requestor) discovers a service (endpoint) and retrieves the service description directly from the service (through meta-data exchange) or from a registry or repository. Service provider and service client roles are logical constructs and a service may exhibit characteristics of both.

Actually, the emergence of new concepts, such as the Internet of Things (IoT), Machine to Machine (M2M), and People to Machine (P2M) caused by the growth of smart devices, allows us to think about the possibility of using all these *things* not only as data producers or consumers, but as services provider. Thus we can think about *internet of things as a cloud of services* software and services composition.

This led to the creation of *mobile providers* which will probably be very different from traditional ESBs; for example they may be context-aware services or location based services, or, given the presence of several sensors which are now equipped mobile devices, providing information from the real world. Moreover, in the context of P2M, they may also act as intermediaries between the people, in order to turn them into *human provided services* [6].

Given the variability of the context, a big challenge is *to automate and efficiently generate a* ***composition of services or software applications*** *that meet exactly the expectations of the requester, which is* ***dependable*** *in time* and distributed.

In the next section we propose a preliminary approach based on immune system, in particular on the antibody component, for supporting dependable and dynamic composition of software and services in the context of the Internet of Things.

4 Dependable Composition of Software and Services in the IoT: A Programmatic Solution

With the popularity and continued growth of smart mobile devices, service providers contained in a composition may be mobile devices. They constantly move and may cause the crash of the composition when, for example, are located outside of the communication range, or one of the device goes offline. The selection of the services, therefore, must make the composition as much as possible *dependable*, so as to avoid frequent recompositions.

In order to achieve a dependable composition it is necessary to provide tools that can handle natively and independently one possible reconfiguration, but above all the ability to adapt to changes in context, while still keeping the goal of the composition. As said in Sect. 3, we can summarize the composition process in:

Service description: is key to service management. Service description enables both the customer and the service provider to know what to expect and not expect from a service. Clearly defined services enable customers to understand service offerings, including what each service does and does not include, eligibility, service limitations, cost, how to request services, and how to get help.

Service classification: once performed discovery of services, the service classification can support the service combination and service selection. For example, it allows to aggregate services discriminating against non-functional requirements, so that one can select the most appropriate service among those identified. It also helps to improve the performance and efficiency of the composition by decreasing the number of services.

Service combination: it is possible to identify two possible approaches: (1) *bottom-up*, which is synthesized in a composition of known services, defining an executable workflow, or (2) *top-down*, where a composer creates abstract non-executable workflow template, and then forwarding the template to the next phase of service selection.

Service Composition Concepts	Immune System Concepts
Service Selection	Memory Cell/Antibody
Service Combination	Immune Response
Service Classification	Macrophage
Service Description	Antigene

Fig. 4. Service and immune system mapping

Immune System Signal	Service Message
Specialization Signal	specializationMessage
Replication Signal	replicationMessage
Activation Signal	activationMessage

Fig. 5. Service message and immune system mapping

Service selection: since a workflow consisting of abstract activity, service selection selecting an appropriate service efficiently for each task. In fact, through the service classification, each activity may have several candidate services with the same functions, that however have, for example, different non-functional requirements.

Since the classification of Fig. 4, we propose a preliminary study in order to provide a tool able to manage efficiently and reliably the **combination** and **selection** of software and services in the context of IoT. To address this challenge, we propose an approach based on the immune system and Functional Programming (FP):

– the elements of the immune system are able to be composed between them in order to build complex structures that once activated have the ability to respond to external stimuli, for example antigens, and exhibit a countermove. This makes it an extremely adaptable, resilient and reliable.
– on the other hand, functional programming encourage the: (1) *compose function*, that favors building programs from the bottom up by composing functions together; (2) *immutability*, since functional programming favors pure functions, which can't mutate data, it also makes use of immutable data. Instead of modifying an existing data, a new one is efficiently created; (3) *transform, rather than mutate, data*, FP uses functions to transform immutable data. One data is put into the function, and a new immutable data structure comes out.

In our view, the services and the software components, are atomic and immutable data that are combined through mechanisms of composition, i.e. functions, in order to ensure the desired goal.

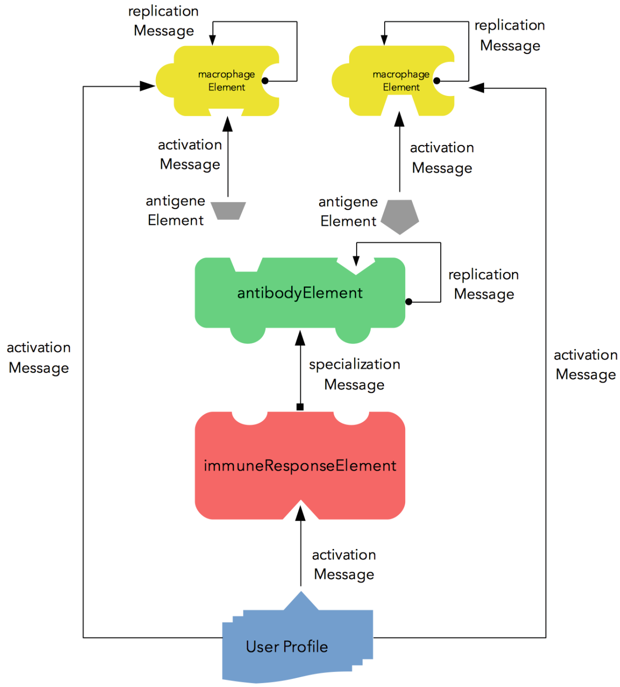


Fig. 6. Antibody architecture

In Fig. 6, we defined the elements constituting the software architecture of the process of Fig. 3(b). Given the highly heterogeneous environment that characterizes the Internet of Things, every actor is equipped with a specific *user profile*, which can be defined as: *a record of user-specific data that define the user's working environment.*

This definition is deliberately general, to ensure the management of profiles that apply to both human and "things", but at the same time *guide* the *classification* and *selection* of services. A further motivation to support the use of a user profile is given by the fact that together with the description of any goal to be achieved, also non-functional requirements can be specified, such as dependability, that for a mobile user is perceived as a fundamental requirement.

The architecture components were represented without *smooth surfaces*, with the intent to represent the ability of each of them to create links in a totally adaptive way, that allows the composition with services or software applications as much automated as possible. The invariant is given by the User Profile, which defines the semantic boundaries, within which to make the classification and the choice of components to be classified.

As introduced in the previous section, we can make some considerations about the formalism and computer science concepts that could support a theoretical implementation of the proposed architecture (Fig. 6). In particular, we consider the concepts of *Partially Applied Functions* [7], where in a function with an parameters list, you can define a new function where you omit one or more parameter of the list.

Let a program $\boldsymbol{p}$, and its input $\boldsymbol{i}$, and we say that we want to split $\boldsymbol{i}$ in a known static part $\boldsymbol{s}$ and a dynamic (unknown) $\boldsymbol{d}$. Then, given a function of specialization $\boldsymbol{S}$, we specialize $\boldsymbol{p}$ with respect to $\boldsymbol{s}$:

$$S(p, < s, \_ >) = p_S$$

In our context, the function S is the service to be provided to the user p, that uses as input parameters: the User Profile (s) and available services at the time (d or $\_$).
This function is computable, and it can be implemented [8]. It is strongly linked to the concept of functional programming. This approach promotes modularity, and thus a natural ability to compose, promoting reuse, parallelization, generality, and automated reasoning. There are numerous programming languages that allow one to adopt the functional programming paradigm; one of these is the Scala language[1].

For simplicity, we map some of the elements of the architecture of Fig. 6 to simple mathematical functions, please see Fig. 7(a) and (b). In particular, we have:

- *map*[2] is $immuneResponseElement$
- s is $UserProfile$
- fsf is $antibodyElement$
- $f1$ and $f2$ is $antigenElement$

```
def fs[X](f:x =>x)(s: Seq[X]) = s map f

def f1(x: int) = x * 2

def f2(x: int) = x * x

def fsf[X](f:x =>x) = fs(f) _

val fsf1 = fsf(f1)

val fsf2 = fsf(f2)
```

(a) Partial Function Mapping

```
Output:

fsf1(List(0,1,2,3)) -> List(0,2,4,6)
fsf2(List(0,1,2,3)) -> List(0,1,4,9)
```

(b) Output

Fig. 7. Example

With this simple example, we have shown how the architecture in Fig. 6 can be mapped in a natural way in a programming language that implements natively functional programming, highlighting the compositional capabilities it offers.

[1] http://www.scala-lang.org.
[2] Scala Language Command - Map works by applying a function to each element in the list.

5 Summary and Conclusions

In this paper we presented a preliminary work with the aim to investigate the opportunity to derive from biological processes software programming concepts that can provide software and services composition solutions in the IoT context. Mimic the behavior of nature is not new in the field of computer science; classical example are neural network [9,10], the genetic algorithm [11], the same immune system [12,13], and more generally the biologically inspired computation [14], they have always evoked great interest. Often, they have characteristics as distributivity, adaptability, heterogeneity, dynamism, dependability, resilience: all new challenges that the advent of mobile devices presents every day.

The interesting aspect of the IT and biology world, it is the fact that some computing paradigms allow you to define in an elegant and formal way some biological processes, such as functional programming, which itself is well supported natively in some modern programming languages like the Scala language. The next steps involve a careful formalization of the proposed architecture (Fig. 6), using the Scala language and the definition of a concrete case study to evaluate the efficiency and fairness of the approach for the construction of systems through dynamic and dependable composition of software and services in the IoT.

Acknowledgment. This research work has been supported by the Ministry of Education, Universities and Research, prot. 2012E47TM2 (project IDEAS - Integrated Design and Evolution of Adaptive Systems), by the European Union's H2020 Programme under grant agreement number 644178 (project CHOReVOLUTION - Automated Synthesis of Dynamic and Secured Choreographies for the Future Internet), and by the Ministry of Economy and Finance, Cipe resolution n. 135/2012 (project INCIPICT - INnovating CIty Planning through Information and Communication Technologies).

References

1. Sundmaeker, H., Guillemin, P., Friess, P., Woelfflé, S. (eds.): Vision and Challenges for Realising the Internet of Things. Publications Office of the European Union, Luxembourg (2010)
2. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (iot): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. **29**(7), 1645–1660 (2013). Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services Cloud Computing and Scientific Applications – Big Data, Scalable Analytics, and Beyond
3. Hofmeyr, S.A.: An interpretative introduction to the immune system. In: Design Principles for the Immune System and Other Distributed Autonomous Systems, pp. 3–26. Oxford University Press (2000)
4. Janeway Jr., C.A., Travers, P., Walport, M., et al.: Immunobiology: The Immune System in Health and Disease. 5th edn. Garland Science, USA (2013)
5. Chappell, D.A.: Enterprise Service Bus - Theory in Practice. O'Reilly, Sebastopol (2004)

6. Li, Q., Liu, A., Liu, H., Lin, B., Huang, L., Gu, N.: Web services provision: solutions, challenges and opportunities (invited paper). In: Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication, ICUIMC 2009, pp. 80–87. ACM, New York (2009)
7. Jones, N.D., Gomard, C.K., Sestoft, P.: Partial Evaluation and Automatic Program Generation. Prentice-Hall Inc, Upper Saddle River (1993)
8. Kleene, S.C.: Introduction to Metamathematics. Ishi Press International, New York (2009)
9. Tahmasebi, P., Hezarkhani, A.: A hybrid neural networks-fuzzy logic-genetic algorithm for grade estimation. Comput. Geosci. **42**, 18–27 (2012). doi:10.1016/j.cageo.2012.02.004
10. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall PTR, Upper Saddle River (1998)
11. Banzhaf, W., Francone, F.D., Keller, R.E., Nordin, P.: Genetic Programming: An Introduction: on the Automatic Evolution of Computer Programs and Its Applications. Morgan Kaufmann Publishers Inc., San Francisco (1998)
12. Kephart, J.O.: A biologically inspired immune system for computers. In: Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp. 130–139. MIT Press (1994)
13. de Castro, L.N.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, London (2002)
14. Parsons, S.: Imitation of life: how biology is inspiring computing by Nancy Forbes, MIT Press, 176 pp., $8.95, ISBN 0-262-06241-0. Knowl. Eng. Rev. **21**, 95–95 (2006)