

Chapter 2

Autonomic Computing and Networking

2.1 Introduction

This chapter presents the rationale behind the introduction of the visionary concept of autonomic computing, as well as its rapidly progressing further translation into and convergence with the domain of modern networked systems. Even though it could be perceived as being more of an introductory, context setting nature only, in fact the chapter provides an extensive, additional commentary and analysis for the purpose of outlining the most comprehensive view on the current status and role of autonomics overall. To this end, not only are the key aspects of the original approach explained but the relevant conceptual and architectural changes are indicated to settle the ground for the introduction of the new concept of Autonomic Cooperative System Architectural Model (ACSAM). In particular, after the general vision has been introduced, the main emphasis is put on the explanation of the workings of the original approach advocating for the transposition of the human Autonomic Nervous System (ANS) into the architecture responsible for the management of complex computer systems. This involves the introduction of the aspects of self-configuration, self-optimisation, self-healing, and self-protection, altogether constituting the notion of self-management. The pertinent architectural assumptions and variations follow and are expanded into the discussion regarding the complementary nature of autonomics and agent systems. Finally, the very issue of convergence is brought up together with the role of self-awareness with regard to the target autonomic networked computer system.

2.2 Vision and Conceptual Correspondence

The emergence of the visionary concept of autonomic computing may be perceived as a sign of the times. Even though the idea of automation has been strived after for a couple of decades, the real outbreak of a tangible advancement in

this field appears to coincide with the beginning of the twenty-first century [10]. Such a situation may be attributed to a stunningly rapid development of highly complex distributed systems, comprising a multitude of interconnected devices, each running complicated software and, thus, posing serious demands in terms of proper configuration and timely maintenance. In fact, initially, the concept of autonomics targeted primarily computer systems, however, as it will be shown throughout the remainder of this book, many crucial architectural variations have been devised thus welding both the realms of computing and networking very closely together [9].

Interestingly, the idea of autonomic computing was inspired the by the way the natural human Autonomic Nervous System operates the body related parameters, such as heart rate or temperature, thereby releasing the brain from controlling them in a fully aware manner [6]. In fact, there is a plethora of such parameters which are considered crucial for the overall functioning of the human organism, yet, a bit surprisingly, the process of control is performed in a somewhat detached and distributed fashion [10]. The organisation of such an autonomic system is said to resemble a recursively arranged hierarchy where each of the numerous self-governing components on a given level encompasses numerous, conceptually similar yet functionally varying, components on a lower level and so forth [6]. In other words, it naturally follows similar patterns understood in a more abstract way, starting from the molecular level, going through human markets and societies, and ending up at the global world's socio-economy [6].

While making an attempt at identifying the reasons for the emergence of autonomics, one may come to a conclusion that, in fact, it is the complexity to serve as its key enabler. In particular, as such complex systems are very demanding in terms of maintenance, there appears to be an urgent need for the application of self-monitoring and self-healing in order to enable the possibility of bypassing their potential malfunctions without any specific human intervention, at least over the majority of time [10]. From the practical point of view, as already mentioned, this issue pertains to the question of the composition of an autonomic computer networked system both in terms of software and hardware. In particular, a system is referred to which encompasses both the self-managing operating systems and intelligent software, as well as redundant memory and processors equipped with self-healing firmware [10]. In this case, however, autonomics is realised through redundancy while, in general, it is especially expected to manifest itself through proper system configuration in the first place.

In fact, nowadays software has become not only a ubiquitous and common commodity but, most of all, a particularly highly complex one. It is so especially for critical applications where any risk of failure may be catastrophic when the consequences are concerned. To mitigate any such effects, certain organisations are claimed to spend even 33–50 % of the Total Cost of Ownership (TCO) of their related computing and communication infrastructure in order to avoid any such software malfunction [5]. It is then believed that only a holistic and systemic approach could be of a value in such a case and, consequently, the concept of

self-managing software is advocated for, calling for self-direction, self-governance, and self-adaptation [5]. This idea is, most obviously, tightly bound with self-managing hardware, where the very same principles of complexity apply, both at the microscopic and macroscopic levels. In particular, this pertains to interconnected computers forming vast and distributed systems, where the notion of self-management interweaves software and hardware.

All in all, usually there are three reasons referred to when the enablers for the advent of autonomic computing are sought for [1]. First of all, the complexity of networked computer systems is mentioned including topics such as highly demanding and complicated database management systems, where the multitude of parameters to be configured has become substantial, if not overwhelming, thus posing certain orchestration issues. Secondly, the commercial development and the resulting considerably vast deployment of Service Oriented Architecture (SOA) adds another dimension of complexity to the above issue, as the distribution of system components often means they are no longer under the command of a single organisation understood in a wide sense. Last but not least comes the very feature of heterogeneity of both software and hardware, not to mention the services offered with their aid. This way the today's networked computing systems are becoming more of a melting pot of an enormous variety of concepts and solutions, thereby calling for the incorporation of the notion of autonomics to facilitate self-management.

The aforementioned issue of autonomic system complexity may become one of its most critical characteristics in terms of proper self-maintenance understood as the, de facto, self-management to be further elaborated on in the following section. Most obviously, such complexity calls for a special treatment and, in fact, it was proposed to be tackled with and orchestrated by the application of an economics and artificial intelligence related mathematical device of utility functions devised to serve as a means of facilitating the specification of preferences [7]. In particular, the utility functions allow to specify a multitude of parameters based on which a learned decision or decisions may be taken by an ADME.¹ By all means does the notion of a learned decision making require the acquisition and processing of the relevant information for the needs of accumulating knowledge [15]. At the same time, the said parameters are claimed to most directly translate into resource-level indicators or, going further, straightforwardly into their related QoS metrics, intended not only to allow for and facilitate the formulation of the optimisation function but, most of all, the identification of the ways and methods of, first, addressing the complexity issue, and, then, solving it, as advocated for in [7].

¹More precisely, originally “rational decisions” of “automated agents” are referred to in this context [7]. The description provided in the book, however, is already targeted towards and adjusted with the theory behind the Generic Autonomic Network Architecture (GANA) to be presented in the next chapter [2].

2.3 Notion and Constituents of Self-Management

In principle, autonomics assumes the exclusion of any human involvement, in fact meaning the one of an administrator, from a rather demanding and time-consuming task of a computer system operation and maintenance. Being able to function without having to be overseen, such a system is then expected to self-manage, thus providing the end users with uninterrupted peak performance [6]. In other words, the system should observe the internal and external conditions, as well as software and hardware issues, and take actions to address them properly. This may include, for example, the process of obtaining software updates, installing them, reconfiguring if necessary, running tests, and, potentially, reverting the previous software version as it may turn out inevitable and necessary in the case of errors [6]. Most precisely, such functionality may be achieved through the use of the following four key characteristic components of the concept of self-management, i.e.: self-configuration, self-optimisation, self-healing, and self-protection, as depicted in Fig. 2.1.

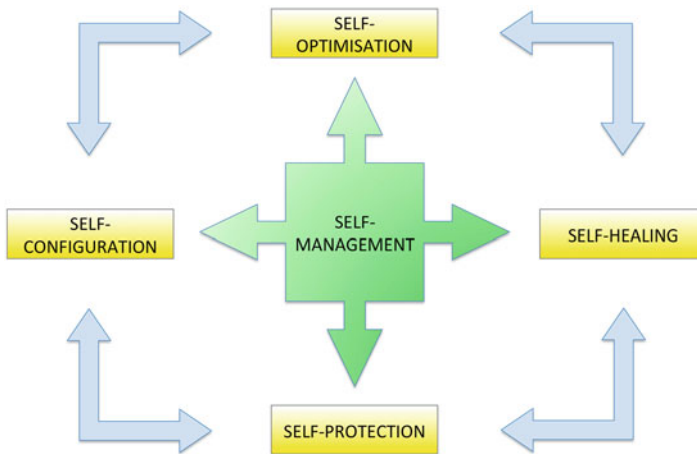


Fig. 2.1 Self-management

In particular, first of all the self-configuration related component is emphasised which pertains directly to the multitude of software and hardware solutions developed and delivered by different vendors, including system elements such as databases, routers, or servers [6]. Given the aforementioned complexity of the currently devised computer systems, the relevant configuration of new equipment may be critically time-consuming, especially when certain unexpected and not easily solvable interoperability issues arise. This aspect becomes especially crucial in the case of widespread set-ups, where the overwhelming number of devices renders manual configuration not only virtually impossible but also highly impractical.

In the first place, an autonomic system is then expected to identify its own capabilities, as well as the properties of a given new component and use some relevant high-level policies for the needs of a proper configuration.

Most obviously, self-configuration goes hand in hand with self-optimisation which should be perceived in terms of parameter tuning, both in the case of software and hardware. Especially, for this very component, the software inherent parameters should not be accidentally confused with the hardware related ones, even though, in both cases, the tuning may be perceived as a purely software driven process. In particular, as the number of parameters is claimed to increase with every system release, the task of self-optimisation similarly appears almost unattainable without the application of some relevantly adjusted autonomic routines [6]. What is more, self-optimisation is not only intended to take place when a new software or hardware component has been attached to the system. Quite the contrary, this process should be continuous as it is the case for the aforementioned Autonomic Nervous System so that modifications could be made on the fly, mostly as a result of being facilitated by certain additionally instrumented learning and reasoning processes and capabilities.

Once the system has carried out and successfully accomplished the above-mentioned tasks of self-configuration and self-optimisation,² it requires to be properly monitored in order to accommodate yet another key feature, i.e. the one related to self-healing. The rationale behind the capability of monitoring stems from the fact that the highly and rapidly increasing complexity of the currently deployed networked computer systems poses certain difficulty in identifying, tracing, and determining the very root cause or causes of a given problem or group of, potentially related, incidents [6]. Non-autonomic approaches may take up to several weeks of a difficult and demanding analysis and result in a highly tangible diagnosis, not to mention the case when the cause or problem has suddenly disappeared [6]. Apparently, such an approach belongs to the resilience related class of solutions where the system might observe certain incidents still before a problem has occurred in order to undertake any relevant action well in advance, thus potentially avoiding otherwise imminent complications [16].

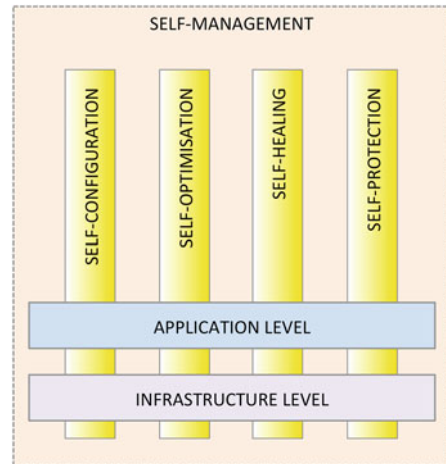
Moreover, the requirement of self-healing is very deeply correlated with self-protection, at the same time exposing certain dose of similarity to fault-management. In fact, due to the mutual relations between resilience and fault-management, the idea of inferring on the basis of symptoms, as indicated previously for the case of self-healing, is undoubtedly maybe even more pertinent to the task of self-protection. In other words, the system is expected to autonomically detect and resolve any unexpected malicious attacks or cascading failures [6]. This way, the data collected during the phase of monitoring may be properly filtered in order to pinpoint the potential root causes and, thus, reason on the yet unknown but sufficiently probable problems, potentially to come. While self-protection should provide the measures necessary for a widely understood failure avoidance, mostly

²In fact, such tasks rather appear to be continuous processes, as the configuration may be changing dynamically and adaptively over time, triggering not only self-optimisation but also self-protection.

consisting in problem mitigation, the routine of self-healing appears to be more of a remedial nature [16]. It means, in turn, that both mechanisms need to interact and base their operation on the aforementioned tasks of self-configuration and self-optimisation.

Obviously, such functionality may only be attained gradually and it is assumed that, initially, all the above-mentioned components would need to be considered and deployed separately. Even though, with the passage of time, self-configuration, self-optimisation, self-healing, and self-protection could become more and more welded together, this process would need to additionally accommodate the evolution of the said self-management understood in a more generic way, as a consistent concept [6]. In other words, first, autonomics would solely help in collecting the data that an administrator could use as a support for the decision making process. Following, the role of autonomic control processes would be expected to be elevated to the level of suggesting certain actions to the human, and, finally, such processes would function in an entirely standalone and detached fashion, basing their own decisions on the actions of some other relevant lower level control processes [6].

Fig. 2.2 Autonomic computing levels



Going further, the presumably horizontal arrangement of the constituents of self-management in the form of self-configuration, self-optimisation, self-healing, and self-protection would need to be perceived from two angles, or, rather, from two levels. In fact, as depicted in Fig. 2.2, the autonomic functions need to be performed at both the application and infrastructure levels [1]. This vertical separation and arrangement appears to be very much in line with the aforementioned close relation, if not in a tangible overlapping, between the realms of computing and networking, where certain aspects advocating for the deployment of automation are, in fact, stemming from the very same root concepts behind the autonomic system design. The main driver for such a separation is the proper maintenance of a sufficient separation between software and hardware, which, on the other hand, may pose

certain additional difficulty, as, nowadays, more and more hardware appears to carry the “software-defined” label.

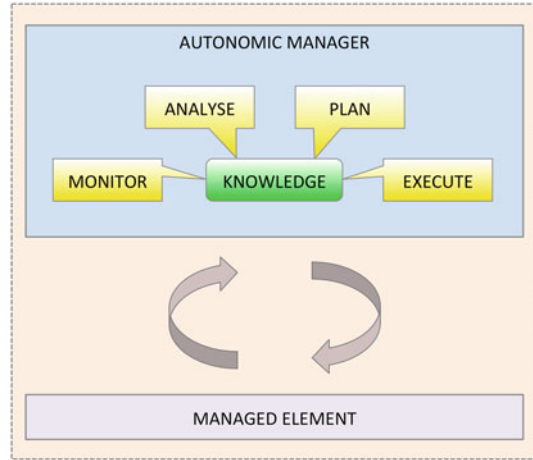
2.4 Architectural Assumptions and Variations

The very initial concept of Autonomic Computing (AC) assumes that a system expressing such a capability should comprise numerous so called Autonomic Elements (AEs), apparently being its most comprehensive, yet not entirely atomic constituents. In particular, AEs are expected to contain resources and deliver services while being able to manage their own behaviour and act pursuant to certain policies, either internal, i.e. imposed by other AEs, or external, i.e. imposed by humans [6]. Given the assumption of a continuous interaction, such an autonomic system is, in fact, claimed to be able to attain the level of a social intelligence of an example ant colony [6]. This does not necessarily impose any particular notion of being cognitive, as such an imitated colony is not a single organism. It is rather a collection of components notifying one another and acting according to directions or rules. As a result, the whole set-up may then work fairly smoothly by adapting itself to the current situation, even though, overall, it presumably lacks any awareness. In other words, the very principle of Autonomic Computing (AC) is reinforced once again by the above example as the ants could be compared to organs of a human body, interacting and exercising behaviour under the umbrella of the same “organism”, of a more virtual nature in this very case, yet retaining its standalone components.

Interestingly, quite the contrary to the later developments aiming to capitalise on the concept of Autonomic Computing (AC), as further elaborated on in Chap. 3, the original solution was proposed under the assumption that the distributed decision making logic would be, in fact, encapsulated within the AEs.³ As depicted in Fig. 2.3, an AE is composed of an Autonomic Manager (AM) and Managed Element (ME) remaining in a very close relation or, rather, interaction through the existence of the control loop which may work at a different pace [18].⁴ It is crucial to note, however, that the inherent feature of each AE, at the same time being its central and focal point, is the notion of knowledge, surrounded by other relevant constituents. In particular, knowledge may be built upon certain pertinent data collected through monitoring and analysis. This is the ability which allows an AE to express some flavour of being able to reason through cognition. While reasoning may be more

³The current designs, such as the Generic Autonomic Network Architecture (GANA), appear to tend to rather expose the internals by modifying the notion of an Autonomic Element when perceived from the global perspective.

⁴Similarly, in the most recent developments the functionality equivalent to an ME is usually referred to as a Managed Entity (ME), while the legacy control loop functions rather as an Autonomic Control Loop (ACL).

Fig. 2.3 Autonomic element

or less limited, it is assumed to manifest itself through tight and proper planning and execution. Planning, however, should incorporate, as its basis, not only the monitoring related data but, preferably, also the aforementioned relevant policies imposed either internally or externally.

Looking into the original design, as it was envisaged in [6], the currently evolved autonomic system architectures, such as the aforementioned GANA, naturally assume that the distinction between AM and ME is highly conceptual. In other words, especially on the higher levels of abstraction, comprising individual computing components, through entire automated enterprises, up to the global economy [6], the difference may be rather intangible, as the MEs could be equivalent to AEs.⁵ Such an equivalence results from the fact that it is possible to take a direct control of any pertinent physical software or hardware components on the lowest level of the hierarchy only. In the majority of cases the entities located above are just intended to encapsulate certain logic which makes them very flexible in terms of the internal structure and the roles they are taking. They are then more containers facilitating the object-orientated system design from the architectural perspective, yet their functionality could be further split, shifted, or translated into some new functional blocks. The lowest level AEs, in turn, need to communicate with their MEs over, preferably standardised [17], interfaces thus making them more rigid, limited, and “hard-coded” in terms of the assumed design [6].

Regardless of the evolution stage of a particular instantiation of the concept of Autonomic Computing, however, it is clearly noticeable that both the original concept and the most recent development in the form of the Generic Autonomic

⁵For the sake of an immediate comparison, the GANA approach, orientated towards distributed networked systems, introduces four conceptual levels given here in the bottom-up order: protocol level, function level, node level, and network level, as introduced in Chap. 3 and further addressed in Chap. 6.

Network Architecture for distributed networked systems are very strongly based on the assumption that an AE needs to manage not only its internal behaviour but also the outside interactions [6]. Such interactions shall mostly translate into the exchange of some relevant signals and messages with other AEs, including the external world [6]. Consequently, the exchanged data is expected to be subject to being processed and issued by the internal logic driven by the objectives embedded into it, as defined by the system architect [6]. Given the existence of vertical, i.e. peering or sibling, and the horizontal, i.e. parental, relations, one may imagine a potentially three-dimensional grid of AEs attempting to maintain equilibrium yet fulfilling certain goal or goals at the same time. This, once again, brings about the reminiscence of the ant colony thus highlighting the very delicate difference between the notion of being autonomic and the notion of being able to reason through thinking. Ants are, in fact, separate organisms moving around together as a symbiotic group which is conceptually quite similar to yet another example of an increased human body temperature that cannot be explicitly driven by the thinking organ, i.e. the brain, but, instead, is controlled by the autonomic behaviour of different organs.

2.5 Agent Systems and Autonomic Entities

As argued in the pioneering work on Autonomic Computing this very concept is claimed to be rather deeply rooted in the theory of agent systems, since autonomy, proactivity, and goal-directed interactivity are the distinguishing characteristics of software agents [6]. What is more, the aforementioned concept of self-management spans not only over single-agent and multi-agent systems, but it also encompasses the rationale behind Service Oriented Architecture (SOA). The above is advocated for on the basis of the comparison between the following definitions referring to both the autonomic computing systems and software agents, respectively, i.e. “computing systems that can manage themselves given high-level objectives from administrators” as opposed to “an encapsulated computer system, situated in some environment and capable of flexible, autonomous action in that environment in order to meet its design objectives”. In fact, both definitions appear to go hand-in-hand and the similarity is obvious, especially that they are cited by undoubtedly one of the milestone publications in the field of autonomic computing, i.e. [6]. Reading into the details one should note, however, that the definition of a software agent clearly mentions the notion of being autonomous but not autonomic. While this might be perceived as a diminishable detail at first sight, after a more thorough analysis it, unfortunately, no longer appears to be so.

More specifically, analysing the issue from a linguistic perspective, the word *autonomous* appears to have two main connotations of relevance to the argument. The definitions may be, in fact, derived immediately from the entries provided by the leading dictionaries which classify the notion of being *autonomous* as having the ability to work and make decisions by itself without any help from anyone else, in

other words, independently [11], or making one's decisions on one's own rather than being influenced by somebody else [3].⁶ First of all, the quality of being autonomous may be then attributed to a system which is of a stand-alone type in the sense that it is self-sufficient and, thus, may operate on its own, without any need for external orchestration, i.e. as a separate entity. Most obviously, the presented perception does not necessarily need to impose the full semantic meaning of autonomics. Secondly, one may think more of a cognitive flavour where the system is autonomous in the sense that it exposes, somewhat, a further advanced quality of being able to reason. Such a quality could manifest itself through the ability to make decisions following some hidden logic, as if the said system was steered by a brain-like device. Quite not surprisingly, also this connotation is certainly not very fortunate as autonomics detaches the features of an Autonomic Nervous System from the learned decisions of the brain it is linked to.

What is more, maybe even a bit unexpectedly, automation of a relevant form and type appears to have been also deployed for the needs of the exploration of the outer space and, consequently, even in that very field one may come across a definite distinction between the notion of autonomics and autonomy. In fact, it is claimed that "while autonomy supports cost-effective accomplishment of mission goals, autonomicity supports survivability of remote mission assets, especially when tending by humans is not feasible" [14]. One should note, however, that in spite of being highly useful, the above citation brings about yet another issue, this time related to the very word "autonomicity", which does not seem to exist in the major dictionaries, if in any at all. This term is currently rather being coined, not only in the cited publication but also in the specifications released by Industry Specification Group on Autonomic network engineering for the self-managing Future Internet (ISG AFI), functioning under the auspices of the European Telecommunications Standards Institute (ETSI).⁷ Both expressions mean automation but here autonomy appears to pertain more to the lack of human control, while autonomics seems to refer to the employment of the principles behind the functioning of the Autonomic Nervous System.

This shows very clearly that even though there is a justified temptation to put both autonomic and agent-driven systems under the very same umbrella, a certain dose of prudence is necessary to avoid any unintended introduction of an unnecessary bias related to the proper understanding of the term "autonomics." What is more, even assuming that there could exist a sufficient dose of correspondence between the two concepts manifesting itself through the employment of individual self-

⁶The connotations outlined by the author come more from relevant academic and industrial discussions, and especially those related to the author's standardisation activity within Industry Specification Group on Autonomic network engineering for the self-managing Future Internet (ISG AFI) of the European Telecommunications Standards Institute.

⁷Unfortunately, according to the author's observations, even though "autonomicity" gains more and more recognition among the experts seeking for the proper expression of their intended meaning, this term appears not to be very well received by the native English language speakers and should be used rather carefully, if at all.

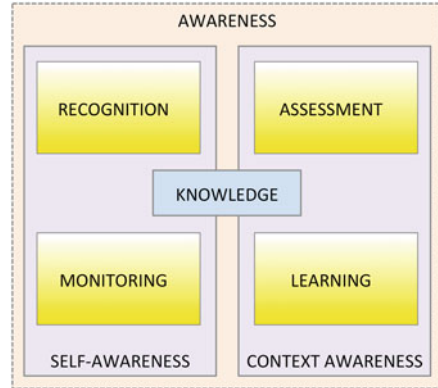
managing Decision Making Elements (DMEs), an autonomic system should be rather analysed from the perspective of Multi-Agent Systems (MASs), as the majority of autonomic systems are composed of multiple sub-systems or services while agent systems comprise multiple agents [1]. This is, in fact, where the Service Oriented Architecture (SOA) related context comes, very neatly, step-by-step, into the global picture of the synergy between the above-mentioned technologies of Autonomic Computing and Multi-Agent Systems. Going further, the autonomic DMEs are claimed to very much resemble software agents in the sense that the latter similarly act on the basis of both the monitoring information obtained from sensors and the imposed policies [1]. Obviously, even though there really appear to exist clear and tight links between both the concepts, one needs to be careful not to warp the sense of autonomic computing by not attributing to autonomics what does not really belong with it.

2.6 Convergence and Self-Awareness

Originally, the process of network management was virtually entirely based on the human-in-the-loop style [4]. The human role was, in fact, exposed to so high an extent that any automation in today's understanding, i.e. organised on a global scale, appeared out of the question or, at least, unattainable, mostly because of the technological obstacles, not to even mention the related deficiency in highly complex architectural developments. This was chiefly changed with the advent of advanced computing systems as devised and introduced by the key stakeholders in the computer market [12]. Consequently, relatively soon after, one could have observed not only a steady but, in fact, an exponential, increase in the incorporation and adoption of the more and more ubiquitous software-based components and entities into the very world of networking. Interestingly, it is actually claimed that should aircraft improve at the same pace as computers, "transatlantic flights would take no more than a few minutes and cost no more than a few dollars" [8].

In fact, as already indicated, the advancement in global convergence between the worlds of computing and networking has most recently reached its apogee, in the positively constructive meaning, and it is becoming more and more difficult to tell the two technological fields apart. The still some years ago rather vivid borders between or among various computer systems connected by telecommunications networks seem to have disappeared and, thus, the hybrid solution may be referred to as, for example, an Autonomic Networked Computing System (ANCS) [13], in order to somewhat artificially emphasise the currently rather passé distinctiveness of both elements, on the one hand, while clearly indicating the actually existing ultimate closeness calling for both the constituents to be perceived as one entity, on the other hand. In fact, it transpires that one of the key drivers for such a high level of convergence is the need of an autonomic networked computing system to be self-contained in the sense of being capable of exposing the ability to behave in a self-aware manner.

Fig. 2.4 Place and role of self-awareness



Self-awareness, in particular, goes hand-in-hand with autonomics, as it assumes the exploitation of the acquired knowledge for the “understanding” of the system status change along with the pertinent implications [15]. Following the classification contained in Fig. 2.4, one may notice, however, that self-awareness is not really a stand-alone feature of an autonomic system, but it is rather complemented by the concept of context awareness. From such a perspective, the context awareness may be defined as the knowledge of a system of how to interact with its surrounding environment through communication and negotiation for the purposes of being able to predict the forthcoming situational states and changes in advance. Most importantly, both self-awareness and context awareness revolve around the above-mentioned focal component denoted as knowledge and are both based on two functional entities, i.e. recognition and monitoring, as well as learning and assessment, respectively [15].

In this respect, the task of recognition is defined to correspond to the exploitation of knowledge for the purposes of tracking changes, while monitoring, most naturally, provides the data necessary for building such knowledge by means of collecting, aggregating, and processing the acquired information [15]. The assessment, in turn, is claimed to be mostly related to testing hypotheses and identifying situational schemata while being responsible for the triggering of learning processes consisting in the formulation of new situational schemata and keeping track of the system evolution history [15]. Based on the cited reference literature one may conclude that there seems to be much more to self-awareness than plain autonomics. In other words, as already indicated, the meaning of autonomics should not be confused with cognition and reasoning, however, while self-awareness clearly involves the processes of learning, it may imply a bit more than simple monitoring for a purely mechanical context recognition.

2.7 Conclusion

In this chapter the rationale behind the concept of autonomic computing and its convergence with modern networked systems was presented. To this end an extensive conceptual analysis was carried out to provide the most comprehensive overview of today's status and role of autonomics. In particular, the general vision and the state-of-the-art in the field of autonomic computer system design was introduced on the basis of the adaptation of the mechanisms driving the behaviour of human Autonomic Nervous System. This involved the discussion of certain aspects of self-configuration, self-optimisation, self-healing, and self-protection, as the chief constituents of the device of self-management. Following, the relevant architectural assumptions and variations were analysed and, then, made subject to the argument regarding the complementary nature of autonomics and agent systems. Finally, the question of convergence between computer and networked systems was investigated to lay the ground for the discussion of the role of self-awareness of the fused architecture of an Autonomic Networked Computing System. Based on this, the novel concept of Autonomic Cooperative System Architectural Model (ACSAM) will be outlined in the next chapter.

References

1. Brazier, F. M. T., Kephart, J. O., Van Dyke Parunak, H., & Huhns, M. N. (2009) Agents and Service-oriented computing for autonomic computing: A research agenda. *IEEE Internet Computing*, 13(3), 82–87.
2. Chaparadza, R., Papavassiliou, S., Kastrinogiannis, T., Vigoureux, M., Dotaro, E., Davy, K. A., et al. (2009) Creating a viable evolution path towards self-managing future internet via a standardizable reference model for autonomic network engineering. In: G. Tselentis, J. Domingue, A. Galis, A. Gavras, D. Hausheer, S. Krco, V. Lotz, & T. Zahariadis (Eds.) *Towards the future internet: A European research perspective*. Amsterdam: IOS Press. ISBN: 978-1-60750-007-0 (Also published at the Future Internet Assembly 2009 in Prague).
3. Collins Cobuild (1997). *Collins Cobuild english dictionary*. New York: HarperCollins.
4. Herrmann, K., Muhl, G., & Geihs, K. (2005) Self management: The solution to complexity or just another problem?. *IEEE Distributed Systems Online*, 6(1), 1–17.
5. Hinchey, M. G., & Sterritt, R. (2006) Self-managing software. *IEEE Computer*, 39(2), 107–109.
6. Kephart, J. O., & Chess, D. M. (2003) The vision of autonomic computing. *IEEE Computer*, 36(1), 41–50.
7. Kephart, J. O., & Das, R. (2007) Achieving self-management via utility functions. *IEEE Internet Computing*, 11(1), 40–48.
8. Mainsah, E. (2002) Autonomic computing: The next era of computing. *Electronics and Communication Engineering Journal* 14(1), 2–3. .
9. Movahedi, Z., Ayari M., Langar R., & Pujolle, G. (2012) A survey of autonomic network architectures and evaluation criteria. *IEEE Communications Surveys and Tutorials*, 14(2), 464–490.
10. Paulson, L. D. (2002) Computer system, heal thyself. *IEEE Computer*, 35(8), 20–22.
11. Pearson Education Limited (2003). *Longman dictionary of contemporary english*. Harlow: Pearson Education Limited, 2003.

12. Pescovitz, D. (2002) Helping computers help themselves. *IEEE Spectrum*, 39(9), 49–53.
13. Phan, C.-V. (2009) Autonomic computing and networking. In *Formal aspects of self-* in autonomic networked computing systems* (pp. 381–410). New York: Springer
14. Truszkowski, W. F., Hinchey, M. G., Rash, J. L., & Rouff, C. A. (2006) Autonomous and autonomic systems: A paradigm for future space exploration missions. *IEEE Transactions on Systems, Man, and Cybernetics Part C (IEEE Transactions on Human-Machine Systems)*, 36(3), 279–291.
15. Vassev, E., & Hinchey, M. (2010) The challenge of developing autonomic systems. *IEEE Computer*, 43(12), 93–96.
16. Wódczak, M. (2011, November). Resilience aspects of autonomic cooperative communications in context of cloud networking. In *IEEE First Symposium on Network Cloud Computing and Applications*. Toulouse, France, pp. 21–23.
17. Wódczak, M., Meriem, T. B., Radier, B., Chaparadza, R., Quinn, K., Kielthy, J., et al. (2011) Standardizing a reference model and autonomic network architectures for the self-managing future internet. *IEEE Network*, 25(6), 50–56.
18. Yixin, D., Hellerstein, J. L., Parekh, S., Griffith, R., Kaiser, G. E., & Phung, D. (2005) A control theory foundation for self-managing computing systems. *IEEE Journal on Selected Areas in Communications*, 23(12), 2213–2222.

Autonomic Computing Enabled Cooperative Networked
Design

Wodczak, M.

2014, XIII, 104 p. 54 illus. in color., Softcover

ISBN: 978-1-4939-0763-2