Autonomic Computing in Cloud: Model and Applications

G. Sobers Smiles David, K. Ramkumar, P. Shanmugavadivu, and P. S. Eliahim Jeevaraj

1 Introduction

The latest trending technologies of the twenty-first century are ubiquitous high-speed networks, smart devices, and edge computing. The beginning stages of the twenty-first century saw the emergence of technologies that allowed mobility, an ability to access Internet and information anytime, anywhere, and anyhow. Cloud computing brought in a fresh wave of computing, where a slice of the resource could be utilized whenever needed and released after its use, bringing down the cost associated with resource management. The best part of cloud computing is that it can deliver both software and hardware as a service on a pay-per-use manner. The services are abstracted in three levels as infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). The benchmarks set by cloud in offering these services are guaranteed service availability and quality, irrespective of the cloud dynamics such as workload and resource variations [1].

Cloud software deployment and service involve configuring large number of parameters for deploying an application, and error-free application configuration is vital for successful delivery of cloud service with quality. This task when done manually resulted in 50% service outages due to human mistakes [2]. The parameters are on the rise and increasingly becoming difficult for a human being

G. Sobers Smiles David · P. S. Eliahim Jeevaraj (✉)
Bishop Heber College, Tiruchirappalli, India
e-mail: eliahimps.cs@bhc.edu.in

K. Ramkumar
Villa College, Male, Republic of Maldives
e-mail: ramkumar.krish@villacollege.edu.mv

P. Shanmugavadivu
Gandhigram Rural Institute – Deemed to be University, Gandhigram, India

T. Choudhury et al. (eds.), *Autonomic Computing in Cloud Resource Management in Industry 4.0*, EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-030-71756-8_3

to keep track. Consider, for Apache servers, there are 240 configurable parameters to be set up and for Tomcat servers, there are more than 100 parameters to be set up and monitored. These parameters are very crucial for the successful running of servers, as they are support files, are related to performance, and are required modules [3]. To be able to find the appropriate configuration for a service, the person in charge of configuring must be thorough in the available parameters and their usage. Complexity is also driven by the diverse nature of the cloud-based web applications.

Chung et al. [4] demonstrated how no single universal configuration is good enough for all the workloads in the web. Zheng et al. [5] showed that it is necessary to update the number of application servers whenever the application server tier was updated in a cluster-based service. It is also important that the systems configuration must also be updated to induct the change in the number of servers. Compared to traditional computing, Cloud computing offers varied level of services but also presents new challenges in application configuration. On-demand hardware resource reallocation and service migration are necessary for this new class of services, and virtualization offers scope for providing these services. Maximizing resource utilization and dynamic adjustment of resources allocated to a VM are the desired features to ensure quality of service. If the application configuration is taken care by automatically and adaptive response for dynamic resource adjustment and reallocation is automatically done, the services could be guaranteed with quality [6]. Autonomic computing is the answer to all the issues mentioned.

The National Institute of Standards and Technology's (NIST) definition of cloud computing [7] has earmarked five essential resource management characteristics for cloud computing. The summary of the characteristics and the requirements are shown in Table 1.

In cloud environment [8], resource management is not a quick process and not an easy task to accomplish. However, the system performance is completely dependent

Table 1 Requirements for dynamic resource management in cloud

	Characteristics	Requirements	Objectives
Cloud computing	On-demand self-service	Intelligent and business-related resources management	Cost optimization and quality of service (QoS) guarantee
	Broad network access	End-to-end resource chain management and resource location optimization	
	Resource pooling	Dynamic deployment management	
	Rapid elasticity	Dynamic adaptive resource management	
	Measured service	Monitoring and reporting of resource usage	

on efficient resource management [9]. In order to provide accurate resource to performance mapping, there are challenges to be overcome. The first challenge is the varying demand of multiple resources due to a mix of hosted applications and its varying workloads [10]. For busy applications, the relationship between performance and resource allocation is inherently nonlinear. The second challenge is performance isolation between coresident applications. Although many existing virtualization techniques such as VMware, KVM, and Xen provide many services like security isolation, environment isolation, and fault isolation, performance isolation is not available. In such scenario, hypervisor deprivation, resource contention could possibly result in performance degradation [11]. The third challenge is to overcome the uncertainty associated with the cloud resources [12]. In the front end, the resources appear as a unified pool, but the background scenario is a complicated process. The Cloud resources are multiplexed, and virtualization of heterogeneous hardware resources is done in the background. Due to this, over a period of time, the actual resources available to hosted applications may vary. Adding to the complexity is the fact that the cloud resource management process is not an independent one. It is interlinked with the management of other layers. A coordinated strategy for configuration management for VM applications is needed. The solutions to resource provisioning issues in cloud computing are summarized in Table 2.

Table 2 Metrics for resource provisioning management in cloud

Name of the Scheme	Functionality
Fault-Tolerant Scheduling [1]	It is dynamic and provides high resource utilization and high schedulability
Online Auction Framework [2]	It follows auction mechanism and provides optimized system efficiency
Meta-heuristic Optimization Technique [13]	It follows particle swarm optimization (PSO) technique and provides minimized workflow execution cost
Agent-based automated service composition (A2SC) [14]	It follows A2SC algorithm and provides reduced VM service cost
Fault Aware Resource Provisioning [15]	It follows FARP algorithm and provides service to redirect the user’
On-Demand Provisioning [16]	It follows augmented shuffled frog leaping (ASFLA) algorithm and provides minimized cost for execution and time
High bandwidth Provisioning [17]	It follows polynomial time energy-aware routing algorithm and provides energy-aware paths to allocate servers and switches for accommodating traffic requests
Fault Tolerance, Fault Detection, and Fault Recovery [18]	Provides self-healing, self-detection, preemptive migration, checkpoint, restart, replication, system node recovery, and job migration techniques
Hybrid Approach [5]	Combination of autonomic computing and reinforcement learning (AC & RL) that could predict the future demands for the cloud services

2 Autonomic Computing Models

Advanced Research Projects Agency (ARPA) is a research body of the Department of Defense, US Government. They research on innovative ways for communication and develop communication systems for the US military. The research done at the ARPA lab is the pioneer research in the modern communication network area. ARPA developed the OSI Model and TCP/IP Model in the late 1950s of the nineteenth century. The whole world of communication follows OSI Model and TCP/IP Model. Similarly, in 1997, ARPA was assigned a project to develop a system with the capability of situational awareness. The system was called "Situational Awareness System (SAS)." As with any other ARPA project, the aim of SAS project was to create devices that aid personal communication and location services for their military personnel. These devices were designed to be used by soldiers on the battlefield. The devices include Unmanned Aerial Vehicles (UAV), Sensors, and Mobile communication systems. The devices could communicate with each other. The data could be collected by all the three devices and shared among them. The latency goal was set below 200 ms [19].

The most important challenge to overcome is that the communication has to take place in enemy's land. This brings two challenges: (1) overcome jamming by the enemy and communicate and (2) minimize interception by the enemy. To overcome the jamming, it was proposed to widen the communication frequency to 20–2500 MHz and bandwidth to 10–4Mbps. In order to overcome interception, it was proposed to employ multi-hop ad hoc routing. In this, every device could send to its nearest neighbor and so on till all the devices are reached. Since the latency was set below 200 ms, the self-management of the ad hoc routing emerged as a huge challenge. The on-field communication process could involve 10,000 soldiers [13].

In the year 2000, ARPA began working on another self-management project [14] named "Dynamic Assembly for System Adaptability, Dependability, and Assurance (DASADA)." This research brought architecture-driven approach in Self-management of Systems. The aim of this project was to research and develop mission-critical systems technology.

Autonomic computing is a term coined by IBM in 2001 to describe computing systems that can manage themselves. Autonomic refers to the automatic reflexes in our body. It is a biological term. In order to overcome the complexities involved with human beings in managing the cloud systems, systems that can manage the functioning of the cloud on their own have emerged. Such systems are referred to as autonomic systems. Autonomic computing includes capabilities such as self-configuration, self-healing, self-optimization, self-protection, and self-awareness to name a few. Autonomic computing approach leads to the development of autonomic software systems [15]. These systems have self-adaptability and self-decision-making support systems to manage themselves in running a huge pool of systems and resources like cloud. While proposing this idea, IBM had laid out specific policies for autonomic computing. They are:

- The system must be aware of its system activities.
- It must provide intelligent response.
- It must optimize resource allocation.
- It must be compatible enough to adjust and reconfigure to varying standards and changes.
- It must protect itself from external threats.

In 2004, ARPA started work on a project titled "Self-Regenerative Systems (SRS)," and its aim was to research and develop military computing systems that could provide services despite damage caused by an attack. In order to achieve this, they proposed four recommendations:

- software with same functionality and different versions
- binary code modification
- Intrusion-tolerant architecture with scalability
- systems to identify and monitor potential attackers within the force.

In 2005, NASA began work on a project titled "Autonomous Nano Technology Swarm (ANTS)." The aim of this project was to develop miniature devices such as "pico-class" spacecraft to do deep space exploration. As the swarm of miniature devices enters the extra-terrestrial boundary, up to 70% of the devices are expected to be lost in the process. With the remaining 30% devices, the exploration has to take place (Fig. 1).

Hence, the devices are organized as colony, where a ruler will give instructions to the remaining devices about the course of action. In addition to the ruler device, a messenger device is planned to take care of the communication between the exploration team and the ground control, such as the round trip delay between the mission control on Earth and the probe device in deep space. NASA planned to

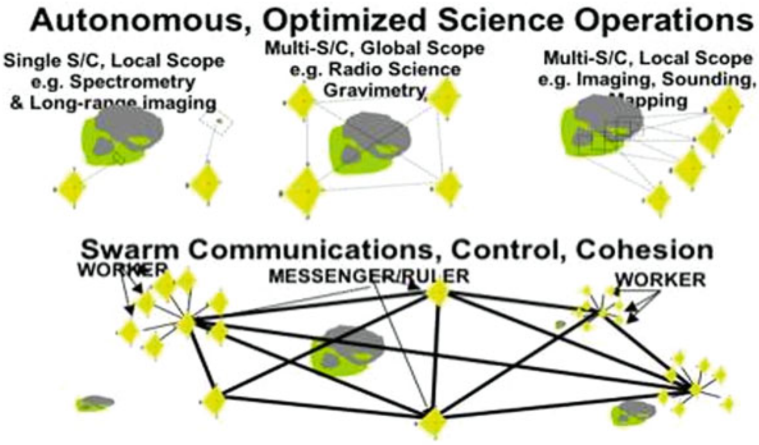


Fig. 1 Autonomous nano swarm

Table 3 Timeline of autonomic computing systems development

Project name	Years	Organization	Features
Situational awareness systems (SA)	1997	ARPA	Self-adaptive network of mobile devices that adjust to varying topology and adjust frequency and bandwidth to varying conditions on the ground.
Dynamic assembly for system adaptability, dependability, and assurance	2000	ARPA	Software systems with ability to probe and gauge for monitoring the system. Adaptation engine to counteract to optimize performance
Autonomic computing (AC)	2001	IBM	Based on human nervous system that employs reflex action. Has four self-management properties: (1) Self-configuring, (2) Self-optimizing, (3) Self-healing, and (4) Self-protecting
Self-regenerative systems (SRS)	2003	ARPA	Self-healing military computing systems that react to attacks or unconditional errors
Autonomous Nano Technology swarm (ANTS)	2007	NASA	The architecture is based on insect colonies. It consists of miniaturized, autonomous, and reconfigurable components for deep space probes

make the devise decide and make decision on its own during critical moments in the probe. Space exploration process strongly needs autonomous systems in order to avert mishaps. These missions are called model-driven autonomic systems [16].

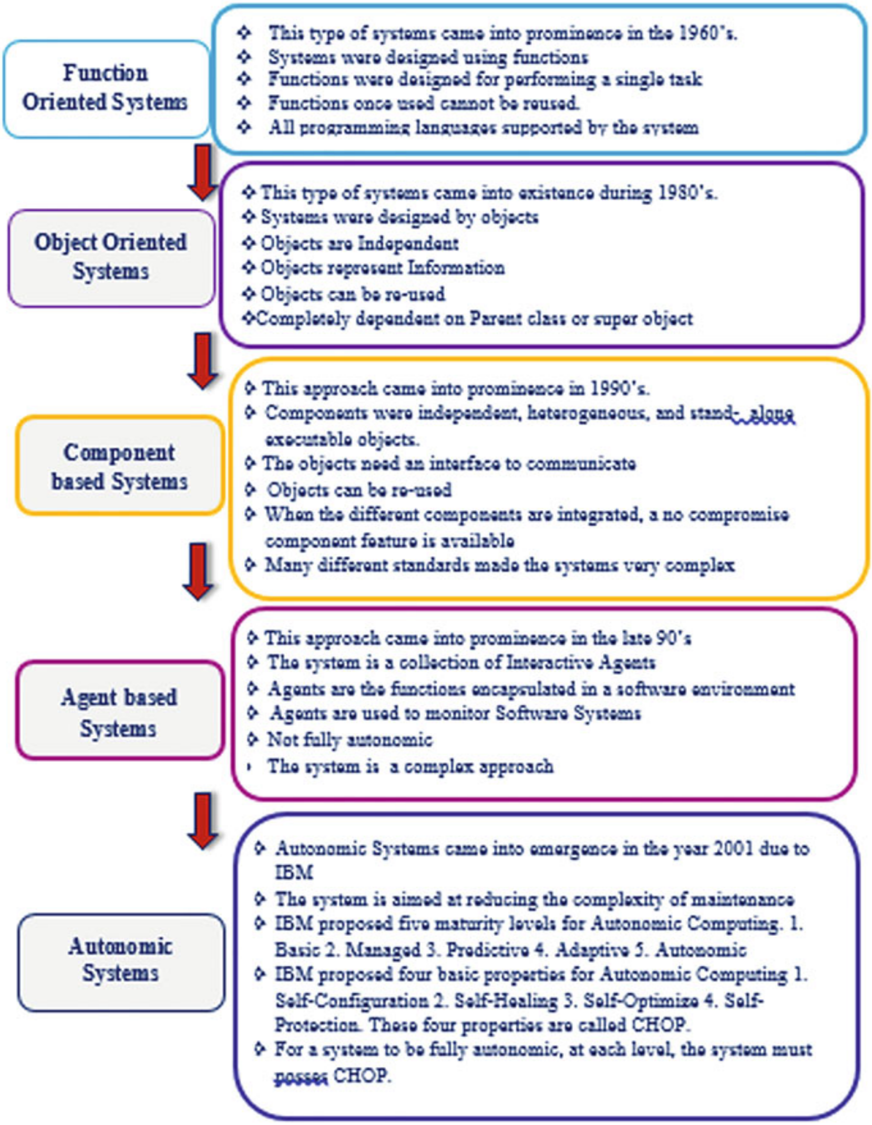
The timeline of the development of autonomic systems is provided in Table 3 [17].

The evolution of autonomic computing is illustrated in Fig. 2 [18]. The evolution started with the function oriented, then object oriented, and component based.

Agent-Based and Autonomic Systems: Automaticity is one of the properties of artificial intelligence. The whole world is witnessing a leap in development and usage of artificial intelligence, and hence, new technologies are emerging in this field [20].

3 IBM's Model for Autonomic Computing

When IBM proposed the concept of autonomic computing in 2001, it had zeroed in on four main self-management properties: (1) self-configuring, (2) self-optimizing, (3) self-healing, and (4) self-protecting. These Self-X properties are inspired by the work of Wooldridge and Jennings (1995) in properties of software agents. They had laid down the following properties for the Self-X management of software systems through software agents [21].



Function Oriented Systems

- This type of systems came into prominence in the 1960's.
- Systems were designed using functions
- Functions were designed for performing a single task
- Functions once used cannot be reused.
- All programming languages supported by the system

Object Oriented Systems

- This type of systems came into existence during 1980's.
- Systems were designed by objects
- Objects are Independent
- Objects represent Information
- Objects can be re-used
- Completely dependent on Parent class or super object

Component based Systems

- This approach came into prominence in 1990's.
- Components were independent, heterogeneous, and stand- alone executable objects.
- The objects need an interface to communicate
- Objects can be re-used
- When the different components are integrated, a no compromise component feature is available
- Many different standards made the systems very complex

Agent based Systems

- This approach came into prominence in the late 90's
- The system is a collection of Interactive Agents
- Agents are the functions encapsulated in a software environment
- Agents are used to monitor Software Systems
- Not fully autonomic
- The system is a complex approach

Autonomic Systems

- Autonomic Systems came into emergence in the year 2001 due to IBM
- The system is aimed at reducing the complexity of maintenance
- IBM proposed five maturity levels for Autonomic Computing. 1. Basic 2. Managed 3. Predictive 4. Adaptive 5. Autonomic
- IBM proposed four basic properties for Autonomic Computing 1. Self-Configuration 2. Self-Healing 3. Self-Optimize 4. Self-Protection. These four properties are called CHOP.
- For a system to be fully autonomic, at each level, the system must posses CHOP.

Fig. 2 Evolution of autonomic computing

- Autonomy:

The software agents can work independently without human intervention and have control over what they do and the internal state.

- Social Ability.

  The software agents can communicate with each other and in some cases with humans through a separate language for agents.

- Reactivity.

  The software agents are reactive in nature and can respond to changes immediately whenever they occur.

- Pro-activeness.

  In addition to being reactive, the software agents can also display proactive behavior before any change occurs.

The concept of autonomic computing was described by Horn [6]. The essence of Autonomic Systems was given by Kephart and Chess [22]. It was a novel attempt to correlate the human nervous system which is autonomic in nature to the select attributes in a computer. Further, the authors proposed autonomic elements (AE) and architecture for autonomic systems (AS). Each autonomic element was proposed to have an autonomic manager with capabilities such as Monitor, Analyze, Plan, and Execute (MAPE) and Knowledge database (K), collectively called as MAPE-K Architecture [22].

IBM's self-management properties are based on the Self-X properties discussed earlier. The self-management properties are dealt in detail by Kephart and Chess (2003) and Bantz et al. (2003) [23].

The IBM's self-management properties are

- Self-configuration.

  Ability to self-configure includes ability to install software according to the needs of the user and the goals.

- Self-optimization.

  Ability to self-optimize includes proactive measures to improve QoS and performance by incorporating changes in the system for optimizing resource management.

- Self-healing.

  Ability to self-heal includes ability to identify faults and problems and take corrective measures to fix the error.

- Self-Protection:

  This ability includes preventing attacks from external as well as internal threats. The external threats can be malicious attacks, and internal threats can be erroneous work by a worker.

The advantages of autonomic computing over traditional computing [23] are summarized in Table 4.

Table 4 Advantages of autonomic computing vs. traditional computing

Autonomic feature	Traditional computing	Autonomic computing
Self-configuration	Installing, configuring, and integrating systems is error-prone and time-consuming	Automated configuration of components and systems follows high-level policies. Rest of the system adjusts automatically and seamlessly
Self-healing	Systems have hundreds of manually set nonlinear tuning parameters and their number increases with each release	Components and systems continuously seek opportunities to improve their own performance and efficiency
Self-optimization	Problem determination in large, complex systems can take a team of programmers weeks	System automatically detects, diagnoses, and repairs localized software and hardware problems
Self-protection	Detection and recovery from attacks and cascading failures is manual	System automatically defends against malicious attacks; it uses early warning and prevents system side failures

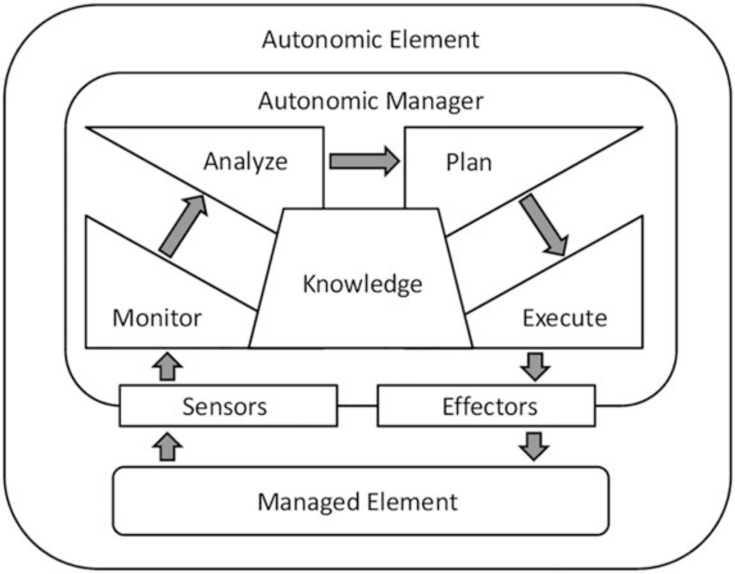


Fig. 3 IBM's MAPE-K architecture reference model

The IBM's self-management properties are incorporated in the IBM's MAPE-K Architecture Reference Model as shown in Fig. 3 [24]. The MAPE-K reference architecture derives its inspiration from the work "Generic Agent Model by Russel and Norvig [2003]." In this model, they had proposed an intelligent software agent that collects data through the sensors. From the data, it infers knowledge and use the knowledge to determine the actions, whichever is necessary.

The key components of the IBM's MAPE-K architecture reference model are [24].

- Autonomic manager
- Managed element
- Sensors
- Effectors
- The managed element

3.1 The Autonomic Manager

It is a software component that collects data through the sensors and can perform to monitor the managed element and, whenever necessary, to execute changes through the effectors. The actions of the autonomic manager are driven by the goals already configured by the administrator. The goals are set in form of event–condition–action (ECA) policy. For example, the goal could be of the form, "When a particular event occurs with a specific condition, then execute a particular action." In such a scenario, conflicts may arise between policies. The autonomic manager applies the knowledge inferred from the internal rules to achieve the goals. Utility functions are also handy when it comes to attain a desired state during a conflict [24]. Further, Nhane et al. proposed the incorporation of an innovative idea like swarm technology in Autonomic Computing. In this technique, the Autonomic Manager is referred to as Bees Autonomic Manager (BAM). Its role is to follow Bee's Algorithm and identify and assign different roles and manage the resource allocation. Further, an exclusive language for autonomic system was proposed by the authors.

3.2 Managed Element

It is a resource that can be a software or hardware that can perform autonomic functions. This autonomic behavior is presented whenever the resource is coupled with autonomic manager.

3.3 Sensors

It is a device that senses the managed element and collects data about it. Sensors or otherwise called as Gauges or Probes. Sensors are used to monitor the resources.

3.4 *Effectors*

Effectors carry out changes to resources or in the resource configuration in response to a situation in computing. There are two types of changes effected, namely, coarse-grained effect where resources are added or removed and thin-grained effect where changes are made to the resource configuration.

4 Challenges in Autonomic Computing

The evolution of autonomic computing has brought some challenges as well. The development of fully autonomic systems is far from over. Such a development needs the challenges to be identified and remedial measures are found out. The challenges include both the coarse-level challenges and fine grain–level challenges [25].

One of the important issues to be addressed is security in autonomic systems. Smith et al. [4] proposed anomaly detection framework to improve the security. Wu et al. [6] proposed an intrusion detection model to improve security in autonomic systems. Nazir et al. [26] proposed an architecture to improve security in Supervisory Control and Data Acquisition (SCADA). Golchay et al. [27] proposed a gateway mechanism between the Cloud and the IoT to improve security. The challenges pertaining to Autonomic Computing comprises of challenges in architecture, challenges in concepts, challenges in the Middle-wares used, and challenges in implementation. IBM has come up with two factors for evaluating the level of autonomicity of fully autonomic systems. They are functionality and recovery-oriented measurement (ROM). Functionality is used to measure the level of dependence on external factors such as human involvement. The recovery-oriented measurement is used to measure the level of availability, scalability, and maintenance. These factors are yet to be applied and measured in any of the autonomic computing systems. At present, factors such as functionality and recovery-oriented measurement are like hypothesis. For an ideal scenario, a fully autonomic system can fulfill the conditions laid out by the two factors (Fig. 4).

5 Applications of Autonomic Systems

Autonomic computing systems find applications in many fields such as smart industry, where large-scale manufacturing is automated, transportation systems where autonomous vehicles are designed and run, healthcare management, Internet of things (IoT), robotics, and 3D Printing [28].

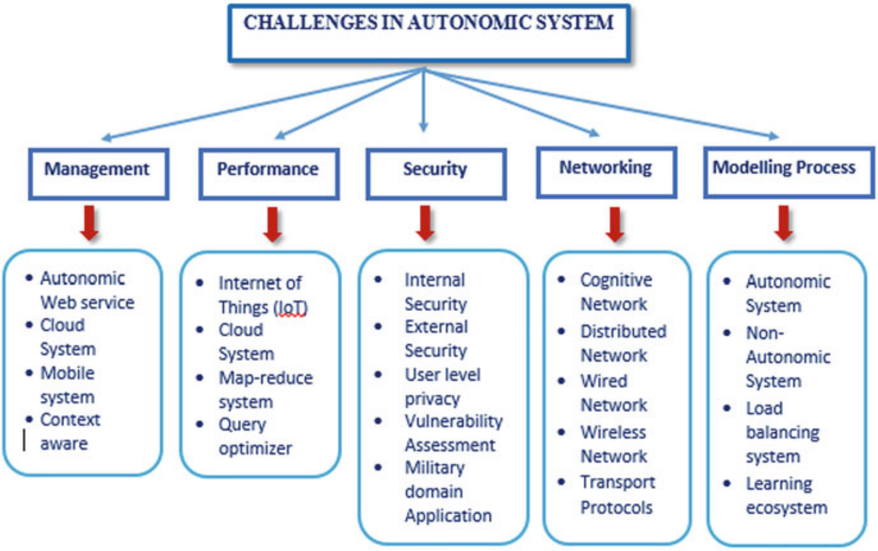


Fig. 4 Challenges in autonomic computing

5.1 *Manufacturing Industry*

Industry 4.0 offers the scope for emerging technologies to be utilized in production systems. This results in maintenance of quality in the products produced and low cost incurred in producing the product. Also, the process of manufacturing is streamlined to result in flawless production and integration of better engineering practices [6]. Manuel Sanchez et al. [23] proposed an integration framework for autonomic integration of various components in Industry 1.0 as shown in Fig. 5.

The framework is designed to monitor the production process and ensure quality. Internet of Services (IoS) and Internet of Everything (IoE) are incorporated in the framework. IoS enhances the communication between various components such as people, things, data, and services involved in the production.

It is proposed that the business process is the managed resource. And hence business process is the service offered (BPaaS). It is proposed to employ everything mining, and hence, people mining, things mining, data mining, and services mining are included. And it is also proposed to include everything mining in autonomic cycles.

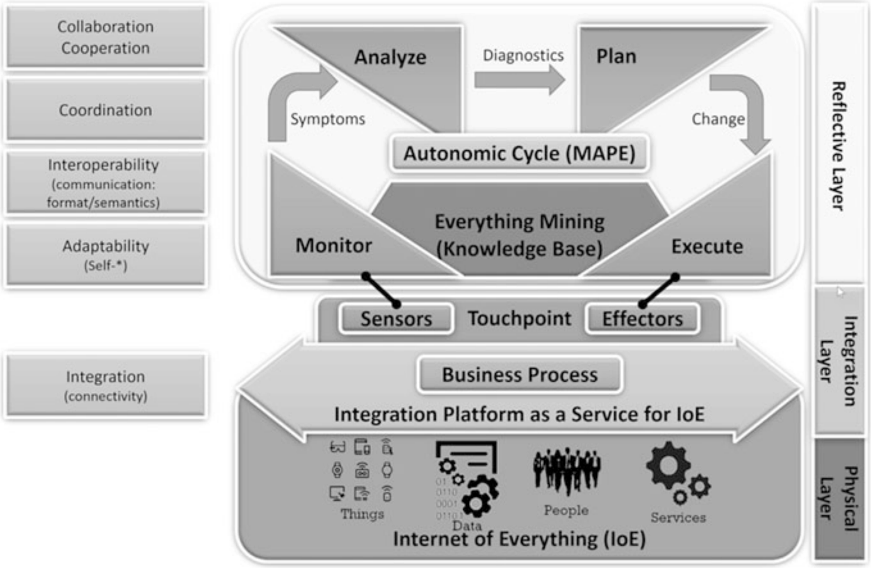


Fig. 5 Integrated framework for autonomic components in industry 1.0

5.2 *Automotive Industry*

One of the main beneficiaries of autonomic computing is the automotive industry. Autonomous vehicles have taken a giant step implementing fully autonomic systems. Every aspect of autonomous vehicle is managed by software [29]. To do so, autonomous vehicles employ machine learning systems, radar sensors, complex algorithms, and latest processors. There are six levels in autonomous vehicles: Fully manual—level 0, single assistance—level 1, partial assistance—level 2, conditional assistance—level 3, high assistance—level 4, and full assistance—level 5. In levels 0, 1, and 2, the human being is responsible for monitoring the driving environment. In levels 3, 4, and 5, the automation system is responsible for monitoring the driving environment. Generally, the vehicles of this class are designed to take orders from the users and hence referred to as automated vehicles rather than autonomous vehicles (Fig. 6).

5.3 *Healthcare Management*

Cloud computing finds application in almost all domains where a large chunk of data are collected and provided using a large pool of resources. Particularly, in remote villages where basic health care is still not available in third-world countries,

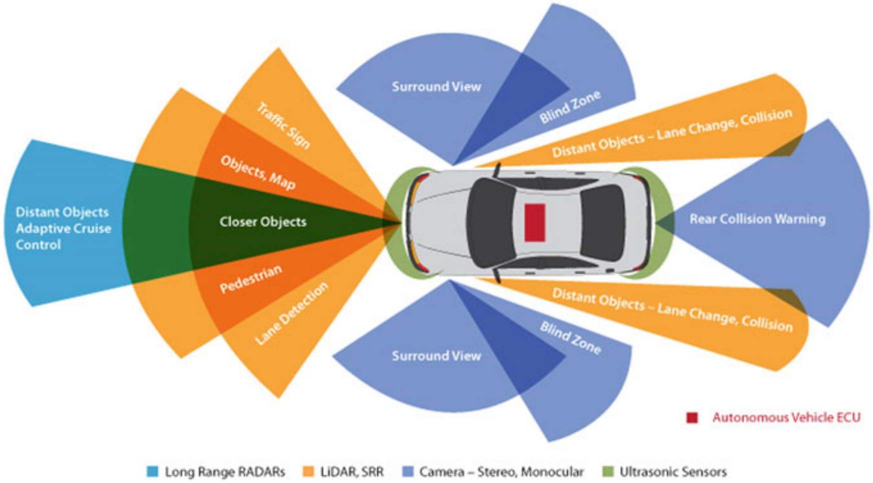


Fig. 6 Automated vehicle

Cloud Computing has provided the opportunity to reach out to the communities and provide better healthcare services [30].

Autonomic systems find application in health care industry in managing complex issues that are otherwise difficult and time-consuming for human beings to operate manually. The services could include pervasive healthcare services. The architecture for autonomic healthcare management is proposed by Ahmet et al. [31]. In this work, the authors have combined the cloud computing and IoT technologies to come up with an autonomic healthcare management architecture.

The efficient healthcare services such as cost-effective and timely critical care could be a reality by incorporating autonomic systems in healthcare management. One bigger bottleneck for developing countries is the population explosion and lack of substantial healthcare professional like Doctors and Technicians. In particular, emergency and trauma care could get the maximum benefits out of autonomic systems in health care (Fig. 7).

5.4 Robotics

With the recent advancements in artificial intelligence and machine learning, robotics field has seen a sea of changes. Robots are now employed in factories performing large-scale operations. Robots are employed in two ways: (1) controlled and (2) autonomous. The controlled robots can perform functions that are instructed or programmed them to do by humans. The successful functioning of a controlled robot includes partial involvement of humans. The autonomous robots can observe the situation and decide to act themselves. No human involvement is need for

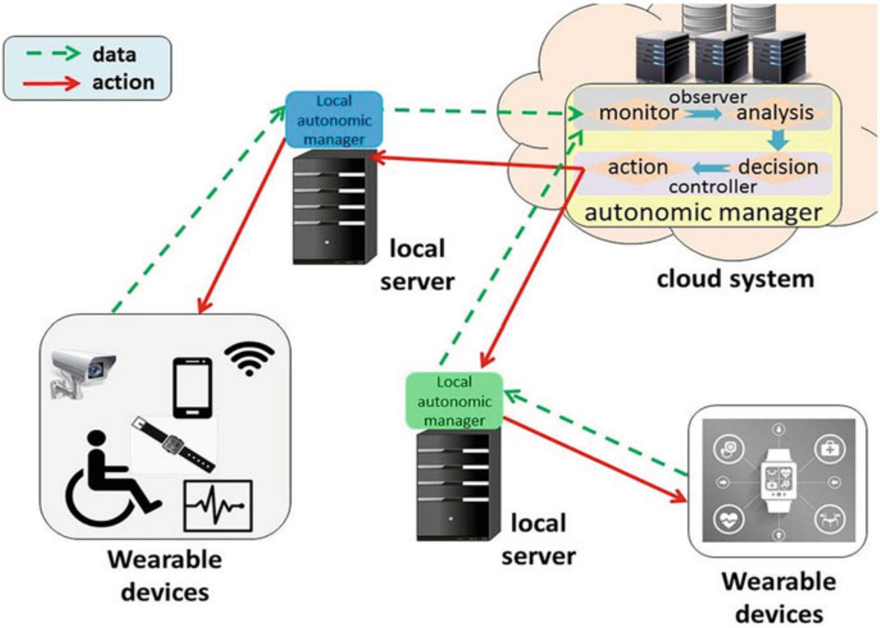


Fig. 7 Autonomic healthcare management architecture

controlled robots [32]. They are intelligent machines. They improve the efficiency and bring down the error. The particular advantage is that they can operate in conditions where human beings cannot like extra-terrestrial explorations [33] (Fig. 8).

6 Conclusion

Autonomic computing has resulted in a new level in IT industry when it comes to automation. Automation has brought down the cost involved and time. Also, the error rate has gone down. These are some of the advantages in the present scenario. In future, as autonomic computing still evolves, it is possible to achieve end-to-end management of services. Further, communication can be made more robust by embedding autonomic capabilities to all the components. The components include Network, Middleware, Storage, Servers, etc. Autonomic systems may be equipped to oversee electricity, transport, traffic control, and in services where the users are more in the future.

Fig. 8 Autonomous robots in space exploration

References

1. Zhu, X., Wang, J., Guo, H., et al. (2016). Fault-tolerant scheduling for real-time scientific workflows with elastic resource provisioning in virtualized clouds. *IEEE Transactions on Parallel and Distributed Systems, 27*(12), 3501–3517.
2. Shi, W., et al. (2016). An online auction framework for dynamic resource provisioning in cloud computing. *IEEE/ACM Trans Network, 24*(4), 2060–2073.
3. Abeywickrama, D. B., & Ovaska, E. (2016). *A survey of autonomic computing methods in digital service eco-systems*. Amsterdam: Springer.
4. Huebscher, M. C., & Mccann, J. A. (2008). A survey of autonomic computing degrees, models and applications. *ACM Computing Surveys.*
5. Ghobaei-Arani, M., Jabbehdari, S., & Pourmina, M. A. (2017). "An autonomic resource provisioning approach for service-based cloud applications: A hybrid approach. *Future Generation Computing Systems*.
6. Wang, W., Jiang, Y., & Wu, W. (2017). Multiagent-based resource allocation for energy minimization in cloud computing systems. *IEEE Trans Syst Man Cybern Syst, 47*, 205–220.
7. Nzanywayingoma, F. (2018). Efficient resource management techniques in cloud computing environment: A review and discussion. *International Journal of Computers and Applications*.
8. Singh, B. K., Alemu, D. P. S. M., & Adane, A. (2020). Cloud-based outsourcing framework for efficient IT project management practices. *(IJACSA) International Journal of Advanced Computer Science and Applications, 11*(9), 114–152.
9. Zheng, Z., Zhou, T. C., Lyu, M. R., King, I., & Cloud, F. T. (2010). A component ranking framework for fault-tolerant cloud applications. Proceedings of the 2010 IEEE 21st international symposium on software reliability engineering (pp. 398–407).
10. Singh, S., Chana, I., & Singh, M. (2017). The journey of QoS-aware autonomic cloud computing. *IT Prof, 19*(2), 42–49.
11. Sobers Smiles David, G., & Anbuselvi, R. (2015). An architecture for cloud computing in higher education. International conference on soft-computing and network security, ICSNS.
12. Tomar, R., Khanna, A., Bansal, A., & Fore, V. (2018). An architectural view towards autonomic cloud computing. *Data Engineering and Intelligent Computing*.

13. Rodriguez, M. A., & Buyya, R. (2014). Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. *Cloud Computing., 2*(2), 222–235.
14. Singh, A., Juneja, D., & Malhotra, M. (2017). A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing. *The Journal of King Saud University Computer and Information Sciences, 29*(1), 19–28.
15. Javadi B, Abawajy J, Buyya R (2012): "Failure-aware resource provisioning for hybrid cloud infrastructure", J Parallel Distributed Computing, 72(10), 1318–1331, (2012).
16. Kaur, P., & Mehta, S. (2017). Resource provisioning and work flow scheduling in clouds using augmented shuffled frog leaping algorithm. *J Parallel Distributed Computing, 101*, 41–50.
17. Yang, S., et al. (2017). Energy-aware provisioning in optical cloud networks. *Computer Networks, 118*, 78–95.
18. Cheraghlou, M. N., Khadem-Zadeh, A., & Haghparast, M. (2016). A survey of fault tolerance architecture in cloud computing. *The Journal of Network and Computer Applications, 61*, 81–92.
19. Dewangan, B. K., Agarwal, A., & Venkatadri, M. (2018). Autonomic cloud resource management. 5th IEEE international conference on parallel, distributed and grid computing (PDGC-2018), *IEEE Digital World.*
20. Kumar, M., & Sharma, A. (2017). An integrated framework for software vulnerability detection, analysis and mitigation: An autonomic system. *Sādhanā, 42*(9), 1481–1493.
21. IBM Corporation. (2005). An architectural blueprint for autonomic computing (3rd ed.).
22. Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *IEEE Computing, 36*(1), 41–50.
23. Lu, Y. (2017). Industry 4.0: A survey on technologies, applications and open research issues. *Journal of Industrial Information Integration, Elsevier Publications.*
24. Doran, M., Sterritt, R., & Wilkie, G. (2018). Autonomic management for mobile robot battery degradation. *International Journal of Computer and Information Engineering, 12*(5), 273–279. ISNI:0000000091950263.
25. Vieira, K., Koch, F. L., Sobral, J. B. M., Westphall, C. B., & de Souza Leão JL (2019). Autonomic intrusion detection and response using big data. *IEEE Systems*.
26. Nazir, S., & Patel, S. (2017). Pat (2017): Autonomic computing meets SCADA security. 16th international conference on cognitive informatics and cognitive computing, ICCI* CC 2017″ (pp. 498–502).
27. Golchay, R., Mouël, F. L., Frénot, S., & Ponge, J. (2011). Towards bridging IOT and cloud services: Proposing smartphones as mobile and autonomic service gateways. *arXiv preprint.*
28. Tahir, M., Ashraf, Q. M., & Dabbagh, M. (2019). IEEE international conference on dependable, autonomic and secure computing. *IEEE Digital Library (2019)* (pp. 646–651).
29. Reduce Automotive Failures with Static Analysis. (2019). www.grammatech.com, Accessed May 20, 2020.
30. Mezghani, E., Expósito, E., & Drira, K. (2017). A model-driven methodology for the design of autonomic and cognitive IOT-based systems: Application to healthcare. *IEEE Transactions on Emerging Topics in Computational Intelligence, Institute of Electrical and Electronics Engineers, 1*(3), 224–234.
31. Ozdemir, A. T., Tunc, C., & Hariri, S. (2017). Autonomic fall detection system. IEEE 2nd international workshops on foundations and applications of self* systems (FAS*W).
32. Puri, G. S., Tiwary, R., & Shukla, S. (2019). A review on cloud computing. *IEEE Computer Society*.
33. Bekey, G. A. (2018). *Autonomous robots from biological inspiration to implementation and control*. Cambridge: The MIT Press.