Volume 4, Issue 1, January 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Autonomic Computing Revolutionizing the way of Managing Complex Systems

T.R.Mudgal
Deptt. Of Physics
CDLM Govt. Engg.
Panniwala Mota, Sirsa, Haryana, India

Susheel Kumar*
Indian Institute of Geomagnetism
College New Panvel,
Navi Mumbai, India

Latika Chaudhary
PVPP College of Engg.
Sion, Mumbai,
India

Y.P.S.Berwal
Additional Director
Technical Education Deptt.
Haryana, India

Abstract*— Autonomic computing is the technology that is building self-managing IT infrastructures i.e. hardware and software that can configure, heal, optimize, and protect itself. By taking care of many of the increasingly complex management requirements of IT systems, autonomic computing allows human and physical resources to concentrate on actual business issues. The term autonomic computing derives from the body's autonomic nervous system, controlling functions like heart rate, breathing rate and oxygen levels without a person's conscious awareness or involvement. The goal is to realize the promise of IT i.e. increasing productivity while minimizing complexity for users. The present paper pursues the goal on many technological fronts as one can actively develop computing systems capable of running themselves with minimal human intervention. Complicated tasks associated with the ongoing maintenance and management of computing systems, autonomic computing technology will allow IT workers to focus their talents on complex, big picture projects that require a higher level of thinking and planning. The ultimate benefit of autonomic computing is freeing IT professionals to drive creativity, innovation and opportunity. Autonomic systems are being created in this manner to recognize external threats or internal problems and then take measures to automatically prevent or correct those issues before human even know there is a problem. These systems are also being designed to manage and proactively improve their own performance, all of which frees IT staff to focus their real intelligence on big picture projects.

***Keywords*— Autonomic Computing, Self Management, Self Monitoring, Self Adjustment, Nervous System, Software.**

I. INTRODUCTION

"Autonomic Computing" is a new vision of computing initiated by IBM. This new paradigm shifts the fundamental definition of the technology age from one of computing, to one defined by data. Access to data from multiple, distributed sources, in addition to traditional centralized storage devices will allow users to transparently access information when and where they need it. At the same time, this new view of computing will necessitate changing the industry's focus on processing speed and storage to one of developing distributed networks that are largely self-managing, self diagnostic and transparent to the user. The term autonomic is derived from human biology. The autonomic nervous system monitors our heartbeat, checks our blood sugar level and keeps our body temperature close to 98.6 °F, without any conscious effort on our part. In much the same way, autonomic computing components anticipate computer system needs and resolve problems with minimal human intervention. However, there is an important distinction between autonomic activity in the human body and autonomic responses in computer systems. Many of the decisions made by autonomic elements in the body are involuntary, whereas autonomic elements in computer systems make decisions based on tasks which are chosen to delegate to the technology. In other words, adaptable policy rather than rigid hard coding determines the types of decisions and actions autonomic elements make in computer systems.

II. KEY ELEMENTS OF AUTONOMIC COMPUTING

1) Knows itself: An autonomic computing system needs to "know itself". Its components must also possess a system identity. Since a system can exist at many levels, an autonomic system will need detailed knowledge of its components, current status, ultimate capacity, and all connections to other systems to govern it. It will need to know the extent of its owned resources, those it can borrow or lend, and those that can be shared or should be isolated.

2) Configure itself: An autonomic computing system must configure and reconfigure itself under varying (and in the future, even unpredictable) conditions. System configuration or setup must occur automatically, as well as dynamic adjustments to that configuration to best handle changing environments.

3) Optimizes itself: An autonomic computing system never settles for the status quo it always looks for ways to optimize its workings. It will monitor its constituent parts and fine tune workflow to achieve predetermined system goals.
4) Heal itself: An autonomic computing system must perform something akin to healing it must be able to recover from routine and extraordinary events that might cause some of its parts to malfunction. It must be able to discover problems or potential problems, then find an alternate way of using resources or reconfiguring the system to keep functioning smoothly.
5) Protect itself: A virtual world is no less dangerous than the physical one, so an autonomic computing system must be an expert in self protection. It must detect, identify and protect itself against various types of attacks to maintain overall system security and integrity.
6) Adapt itself: An autonomic computing system must know its environment and the context surrounding its activity, and act accordingly. It will find and generate rules for how best to interact with neighbouring systems. It will tap available resources, even negotiate the use by other systems of its underutilized elements, changing both itself and its environment in the process in a word, adapting.
7) Open itself: An autonomic computing system cannot exist in a hermetic environment. While independent in its ability to manage itself, it must function in a heterogeneous world and implement open standards in other words, an autonomic computing system cannot, by definition, be a proprietary solution.
8) Hide itself: An autonomic computing system will anticipate the optimized resources needed while keeping its complexity hidden. It must marshal IT resources to shrink the gap between the business or personal goals of the user, and the IT implementation necessary to achieve those goals without involving the user in that implementation.
In an autonomic environment, system components from hardware such as desktop computers and mainframes to software such as operating systems and business applications are self-configuring, self-healing, self-optimizing and self-protecting.

III. FUNCTIONAL CHARACTERISTICS OF AUTONOMIC COMPUTING

In a self-managing Autonomic System, the human operator takes on a new role. He does not control the system directly. Instead, he defines general policies and rules that serve as an input for the self-management process. For this process, IBM has defined the following four functional characteristics:
1) Self configuration: Adapt automatically to the dynamically changing environment. Internal adaptation systems add/remove new components and configure itself on the fly. External adaptation systems configure themselves into a global infrastructure. The seamless integration of new hardware resources and the cooperative yielding of resources by the operating system is an important element of self configuring systems. Hardware subsystems and resources can configure and re-configure autonomously both at boot time and during run time. This action may be initiated by the need to adjust the allocation of resources based on the current optimization criteria or in response to hardware or firmware faults. Self configuring also includes the ability to concurrently add or remove hardware resources in response to commands from administrators, service personnel, or hardware resource management software.
2) Self healing: Discover, diagnose and react to disruptions without disrupting the service environment. Fault components should be detected, isolated, fixed and reintegrated. With self-healing capabilities, platforms can detect hardware and firmware faults instantly and then contain the effects of the faults within defined boundaries. This allows platforms to recover from the negative effects of such faults with minimal or no impact on the execution of operating system and user level workloads.
3) Self optimizing: Monitor and tune resources automatically. Support operating in unpredictable environment, efficiently maximization of resource utilization without human intervention. Self optimizing capabilities allow computing systems to autonomously measure the performance or usage of resources and then tune the configuration of hardware resources to deliver improved performance.
4) Self protection: Anticipate, detect, identify and protect against attacks from anywhere. Defining and managing user access to all computing resources, Protecting against unauthorized resource access, e.g. SSL (Secure Socket Layer), Detecting intrusions and Reporting as they occur. This allows computing systems to protect against internal and external threats to the integrity and privacy of applications and data.

IV. IBM AUTONOMIC FRAME WORK

Applications of autonomic services in the enterprise are increasing day by day in all types of networked environments. With an ever-increasing number, type, and complexity of network services available to individual computing systems comes an increasing complexity in establishing and maintaining the configurations of these services. Many network services are already self configuring today, but this capability is not yet universally available for the broad spectrum of network services or networked environments. Without wide reaching network services self configurability, the benefits of reduced management complexity will remain unrealized. We begin by examining existing network services configuration technologies and identifying incomplete or inconsistent capabilities for dynamically self configuring network services. We present for consideration the requirements of an architecture for dynamically self configuring network services that drives enhanced yet simplified capabilities both to end users and to IT technicians and engineers in a corporate IT environment, as well as to roaming wireless users and home networks. We then continue by examining the practical implementation of autonomic network service configuration. Purely autonomic systems cannot easily be built today due to a lack of comprehensive framework support. However, substantial pieces of autonomic technology exist in forms suitable for early adoption. Specifically, the focus is on the IBM Autonomic Computing Toolkit, an open set of

Java-class libraries, plug-ins, and tools created for the eclipse development environment. The IBM autonomic computing toolkit represents a modern framework for enterprise software integration. In present article the toolkit's standard interfaces and data formats are examined to identify its applicability to network services configuration problems and a summary of findings and recommendations for prospective enterprise developers and integrators of autonomic tool kits is concluded.

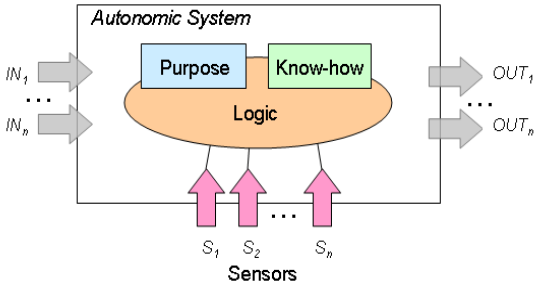

Fig.1 Conceptual Model of Autonomic Computing

V. The IBM Autonomic Frame Reference Model

1) Autonomic manager: The autonomic manager is a component that implements the control loop. The autonomic manager distinguishes the autonomic element from its non autonomic counterpart. By monitoring the managed element and its external environment, and constructing and executing plans based on an analysis of this information, the autonomic manager will relieve humans of the responsibility of directly managing the managed element. The architecture dissects the loop into four parts that share knowledge:

2) Monitor: The monitor part provides the mechanisms that collect, aggregate, filter, manage and report details (metrics and topologies) collected from an element.

3) Analyse: The analyse part provides the mechanisms to correlate and model complex situations (time-series forecasting and queuing models, for example). These mechanisms allow the autonomic manager to learn about the IT environment and help predict future situations.

4) Plan: The plan part provides the mechanisms to structure the action needed to achieve goals and objectives. The planning mechanism uses policy information to guide its work.

5) Execute: The execute part provides the mechanisms that control the execution of a plan with considerations for on-the-fly updates.

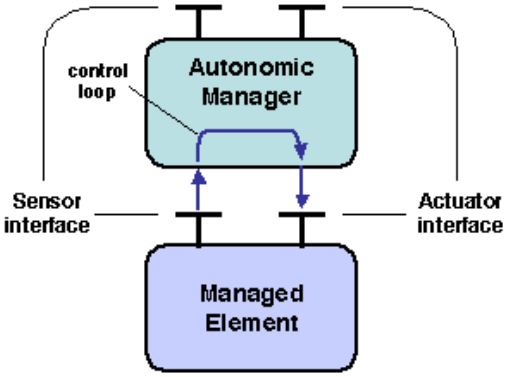

Fig.2 Autonomic Reference Model

6) Control loops: A control loop can be provided by a resource provider, which embeds a loop in the runtime environment for a particular resource. In this case, the control loop is configured through the manageability interface provided for that resource (for example, a hard drive). In some cases, the control loop may be hard wired or hard coded so it is not visible through the manageability interface. Autonomic systems will be interactive collections of autonomic elements individual system constituents that contain resources and deliver services to humans and other autonomic elements. An autonomic element will typically consist of one or more managed elements coupled with a single autonomic manager that controls and represents them. In an autonomic environment, autonomic elements work together; communicate with each other with high level management tools. They regulate themselves and, sometimes, each other. They can proactively manage the system, while hiding the inherent complexity of these activities from end users and IT professionals. Another aspect of the autonomic computing architecture is shown in the diagram below. This portion of the architecture details the functions that can be provided for the control loops. The architecture organizes the control loops into two major elements, a managed element and an autonomic manager. A managed element is what the autonomic manager is controlling. An autonomic manager is a component that implements a control loop in an autonomic computing architecture, control loops facilitate system management.

7) Managed elements: The managed element is a controlled system component. The managed element essentially is equivalent to what is found in element with ordinary non-autonomic systems, although it can be adapted to enable the autonomic manager to monitor and control it. The managed element could be a hardware resource, such as storage, CPU, or a printer, or a software resource, such as a database, a directory service, or a large legacy system. At the highest level, the managed element could be an e utility, an application service, or even an individual business. The managed element is controlled through its sensors and effectors.

8) Sensors: The sensors provide mechanisms to collect information about the state and state transition of an element. To implement the sensors, one can either use a set of "get" operations to retrieve information about the current state, or a set of management events (unsolicited, asynchronous messages or notifications) that flow when the state of the element changes in a significant way.

9) Effectors: The effectors are mechanisms that change the state/configuration of an element. In other words, the effectors are a collection of "set" commands or application programming interfaces that change the configuration of the managed resource in some important way. The combination of sensors and effectors form the manageability interface is available to an autonomic manager. As shown in the figure above, by the black lines connecting the elements on the sensors and effectors sides of the diagram, the architecture encourages the idea that sensors and effectors are linked together. For example, a configuration change that occurs through effectors should be reflected as a configuration change notification through the sensor interface.

VI. BENEFITS OF AUTONOMIC COMPUTING

Autonomic computing was conceived to lessen the spiraling demands for skilled IT resources, reduce complexity and to drive computing into a new era that may better exploit its potential to support higher order thinking and decision making. Immediate benefits will include reduced dependence on human intervention to maintain complex systems accompanied by a substantial decrease in costs. Long term benefits will allow individuals, organizations and businesses to collaborate on complex problem solving.

1) Short term IT related benefits: Short term few benefits may be simplified user experience through a more responsive, real-time system, cost savings scale to use, scaled power, storage and costs that optimize usage across both hardware and software, full use of idle processing power, including home PC's, through networked system. Natural language queries allow deeper and more accurate returns and seamless access to multiple file types. Open standards will allow users to pull data from all potential sources by re-formatting on the fly. Stability, high availability, high security system, fewer system or network errors due to self healing and improved computational capacity are also the short term IT related benefits of autonomic computing.

2) Long term/ higher order benefits: To realize the vision of enablement by shifting available resources to higher order business, embedding autonomic capabilities in client or access devices, servers, storage systems, middleware, and the network itself, constructing autonomic federated systems and achieving end-to-end may be the long term benefits of autonomic computing.

3) Service level management: The service level management is accelerated implementation of new capabilities, collaboration and global problem solving. Distributed computing allows for more immediate sharing of information and processing power to use complex mathematics to solve problems. Massive simulation such as weather, medical complex calculations like protein folding, which require processors to run 24x7 for as long as a year at a time.

VII. CHALLENGES FOR AUTONOMIC COMPUTING

*1) System identity***:** Before a system can transact with other systems it must know the extent of its own boundaries. How will one design his systems to define and redefine themselves in dynamic environments?

2) Interface design: With a multitude of platforms running, system administrators face a, How will we build consistent interfaces and points of control while allowing for a heterogeneous environment?

3) Translating business policy into IT policy: The end result needs to be transparent to the user. How will we create human interfaces that remove complexity and allow users to interact naturally with IT systems?

*4) Systemic approach***:** Creating autonomic components is not enough. How can we unite a constellation of autonomic components into a federated system?

5) Standards: The age of proprietary solutions is over. How can we design and support open standards that will work?

*6) Adaptive algorithms***:** New methods will be needed to equip our systems to deal with changing environments and transactions. How will we create adaptive algorithms to take previous system experience and use that information to improve the rules?

7) Improving network monitoring functions: To protect security one has to detect potential threats and achieve a level of decision making that allows for the redirection of key activities or data. Smarter microprocessors can be used to detect errors and anticipate failures.

VIII. FUTURE SCOPE

Some of the current components and their proposed development under autonomic computing, includes SMS, SNMP, adaptive network routing, network congestion control, high availability clustering, ESS, RAID, DB optimizer, virus management. In case of SMS, its level of sophistication is serving the world (i.e., people, business processes) used for Policy management and storage tank i.e. policy managed storage for every file or folder, the user sets policies of availability, security, and performance. The system figures out where to put the data, level of redundancy and level of backup. This indicates that autonomic computing is goal oriented management. Its Future goal is Policy language and

protocols. The level of sophistication of SNMP is heterogeneous components interaction which is used for workload management. Its Future goal is autonomic computing stack, social policy and DB/storage co-optimization. Adaptive network routing, network congestion control, high availability clustering has a level of sophistication of homogeneous components interacting. It is used for collective intelligence, storage bricks, the idea is to have higher redundancy than RAID, protection of performance hot spots with proactive copies and elimination of repair for life of system by building extra drives into the system. Its Future goal is new packaging concepts for storage i.e. to change the packaging of an array of disks from a 2-D grid to a 3-D cube. There is a prototype of this called the Ice Cube which is basically the size of a medium sized packing box with size upto 1 Peta byte (10^{15} bytes), 250kW, 75dB air noise which should last for 5 years without any service ever. Other components include ESS, RAID, DB optimizer, virus management used for eLiza, SMART/LEO (Learning in Query Optimization), and software rejuvenation.

IX. CONCLUSIONS

The autonomic concept has been adopted by today's leading vendors and incorporated into their products. Aware that success is tied to interoperability, many are participating in the standards development necessary to provide the foundation for self-managing technological ecosystems, and are integrating standards into their technology. IBM is making a substantial investment in the autonomic concept and has released its first wave of standards-based components, tools and knowledge capital. IBM offers a wide array of service offerings, backed by methodology and tools, which enable and support the adoption of Autonomic Computing. Autonomic capabilities are critical to businesses with large and complex IT environments, those using Web Services and/or Service Oriented Architecture (SOA) models, and those that leverage. These are also key enablers for smaller businesses seeking to take advantage of current technologies, because autonomic computing helps mask complexity by simplifying infrastructure management, business or e-commerce.

REFERENCES

[1] Xiaolong Jin and Jiming Liu, "From Individual Based Modelling to Autonomy Oriented Computation", in Matthias Nickles, Michael Rovatsos, and Gerhard Weiss (editors), *Agents and Computational Autonomy: Potential, Risks, and Solutions*, pages 151–169, Lecture Notes in Computer Science, vol. 2969,Springer, Berlin, 2004.
[2] Autonomic Computing: IBM's Perspective on the State of Information Technology, http://www.research.ibm.com/autonomic/manifesto/autonomic_computing.pdf.
[3] 'Trends in technology', survey, Berkeley University of California, USA, March 2002.
[4] IEEE Computer Magazine, Jan 2003.
[5] S-Cube Network, "Self-Healing System".
[6] E. Curry and P. Grace, "Flexible Self-Management Using the Model-View-Controller Pattern,".

