Autonomic Computing

Shama Wankhade, Payal Koshatwar, Rupali Thakare

* Department of Computer Science and Engineering, JDIET Yavatmal

Abstract- When we work harder, our hearts beat faster. When we're hot, we sweat. The internal functions within our body regulate themselves. Correspondingly, won't we like to have systems that can heal themselves? Autonomous Computing is a holistic, system-wide perspective. The ultimate goal is to have the machine think for itself without human input, to be able to boot up backup systems, and even to order spare parts to ensure transparency to the user. Successful autonomic systems will need to be self-configuring, self-optimizing, self-protecting, and self-healing. So rather than having to tell systems what to do explicitly in a certain situation, we just tell these systems what we're trying to accomplish. While that ultimate goal is still years away, the first generation of self-adapting software tools is within reach. Autonomic computing will not reinvent computer science but will offer new opportunities—new ways of securing our systems against attack.

Index Terms- self-configuring, self-optimizing, self-protecting, self-healing

I. INTRODUCTION

Autonomic computing is an approach to self-managed computing systems with a minimum of human interference. The term derives from the body's autonomic nervous system. The human body is self-healing. Broken bones mend, cuts heal, and a child's immunity system grows stronger. The body's autonomic nervous system, which controls involuntary actions without conscious awareness or involvement, has fascinated the world of medicine. So why can't it be the same with computers? Must a computer engineer or a systems administrator monitors a server round-the-clock to ensure normal operation? The solution is autonomic computing systems that will have the ability to configure, tune and even repair themselves. Machines that analyze the difficulties and re-route to alternate ultimate networks, computers or even switch to a backup chip. The goal is to have the machine think for itself without human input, to be able to boot up backup systems, and even to order spare parts to ensure sure that the people running the system never know there is a problem.

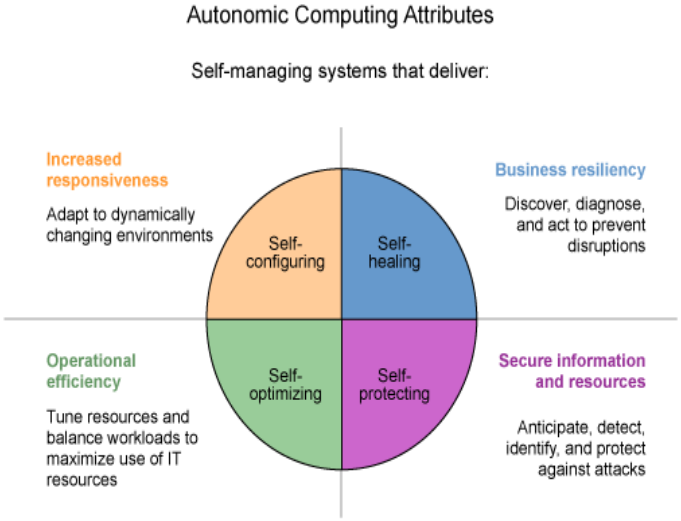
II. ARCHITECTURE FEATURE

Autonomic computing will have implications for computing systems at all scales, from single devices to the worldwide networked economy. One very common architecture for an autonomic element will involve two parts:

- ***Functional unit:*** that performs whatever basic function the element provides (such as storage, database functions, Web services, and so on)
- ***Management unit:*** that oversees the operation of the functional unit, ensures that it has the resources that it needs to perform its function, configures and reconfigures it to adapt to changing conditions, carries out negotiations with other autonomic elements.
- What is "Self-CHOP?"

The acronym CHOP is shorthand for configure, heal, optimize, and protect, the fundamental aspects of autonomic computing technology. Autonomic systems are designed to address one or more of these aspects. Figure 1 illustrates these concepts.

Figure. Self-CHOP



An architectural blueprint for autonomic computing explains the self-CHOP attributes:

Although autonomic, control loops consist of the same fundamental parts, their functions can be divided into four broad embedded control loop categories.

- Self-configuring - It Can dynamically adapt to changing environments. Self-configuring components adapt dynamically to changes in the environment, using policies provided by the IT professional. Such changes could include the deployment of new components or the removal of existing ones, or dramatic changes in the system characteristics.
- Self-healing - It Can discover, diagnose and react to disruptions. Self-healing components can detect system malfunctions and initiate policy-based corrective action without disrupting the IT environment. Corrective action could involve a product altering its own state or

effecting changes in other components in the environment.

- Self-optimizing - It Can monitor and tune resources automatically. Self-optimizing components can tune themselves to meet end-user or business needs. The tuning actions could mean reallocating resources -- such as in response to changing workloads -- to improve overall utilization, or ensuring that particular business transactions can be completed in a timely fashion. Self-optimization helps provide a high standard of service for both the system's end users and a business's customers.
- Self-protecting - It Can anticipate, detect, identify and protect against threats from anywhere. Self-protecting components can detect hostile behaviors as they occur and take corrective actions to make themselves less vulnerable. The hostile behaviors can include unauthorized access and use, virus infection and proliferation, and denial-of-service attacks. Self-protecting capabilities allow businesses to consistently enforce security and privacy policies.

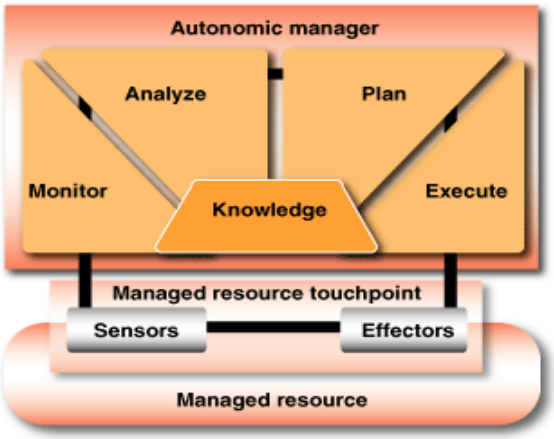
Ex. ***ABS (Anti-lock braking system)***

-Safety innovations by engineers

1. ABS prevents the wheel from locking when the car goes into a skid. This ensures the car can still be steered and thus prevents accidents. The ABS comprises electronic sensors and solenoid valves in the wheel hubs.
2. Windows XP also incorporates self-healing technology. When an application crashes, the user can shut it down systematically, thereby preventing the entire system from freezing or hanging. This operating system also offers to report program errors to the Microsoft Support team. Further, Windows XP looks out for updates and automatically downloads these when available.
3. Recent versions of Microsoft Office include a Repair feature. So if key program file (such as Winword.exe) gets corrupted or accidentally deleted, the software can reinstall it. Such features will soon be present in other desktop software.

A good example would be RAID, which allows a storage system to have a disk drive fail and it just keeps running nicely, it basically works around and heals the failure of that disk drive.

III. AUTONOMIC CYCLE



Autonomic manager

The autonomic manager is a component that implements the control loop. The architecture splits the loop into four parts that share knowledge. They are:

- Monitor--provides the mechanisms that collect, aggregate, filter, manage, and report details (metrics and topologies) collected from a managed resource
- Analyze--provides the mechanisms to correlate and model complex situations that allow the autonomic manager to learn about the IT environment and help predict future situations
- Plan--provides the mechanisms that construct the action needed to achieve goals and objectives
- Execute--provides mechanisms that control the execution of a plan with considerations for dynamic updates

The four parts work together to provide the control loop functionality. They consume and generate knowledge. This knowledge builds on known information about the system and grows as the autonomic manager learns more about the characteristics of the managed resources. The knowledge is continuously shared among the four parts, leading to more informed decisions being made by the parts. It shows a structural arrangement of the parts--not a control flow. The bold line that connects the four parts should be thought of as a common messaging bus rather than a strict control flow. In other words, there can be situations where the plan part might ask the monitor part to collect more or less information. There could also be situations where the monitor part may trigger the plan part to create a new plan.

IV. MANAGED RESOURCE

The managed resource is a controlled system component. There can be a single managed resource (a server, database server, or router) or a collection of resources (a pool of servers, cluster, or business application).An autonomic manager communicates with a managed resource through the manageability interface. A touchpoint is the implementation of the manageability interface by a specific managed resource. For example, a database server might implement a touchpoint for communicating with an autonomic manager.

V. MANAGED RESOURCE TOUCHPOINT

The touchpoint, one of the three main elements of the autonomic computing architecture, delivers the manageability interface to the autonomic manager. The manageability interface between an autonomic manager and a managed resource is organized into sensor and effector operations. In the simplest terms, sensor operations are typically used to transmit events or properties to an autonomic manager, whereas effector operations are typically used to cause some sort of change in a managed resource, such as altering state data or setting property values.

The combination of sensor and effector operations forms the manageability interface that is available to an autonomic manager. As shown in Figure , by the black lines connecting the sensor and effector sides of the diagram, the architecture encourages the idea that sensor and effector operations are linked together.

VI. BENEFITS

Autonomic computing was conceived to lessen the spiraling demands for skilled I/T 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.

I/T related benefits:

- Cost-savings - scale to use.
- Full use of idle processing power, including home PC's, through networked system.
- Natural language queries allow deeper and more accurate returns.
- 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.

Higher Order Benefits:

- Embedding autonomic capabilities in client or access devices, servers, storage systems, middleware, and the network itself. Constructing autonomic federated systems.
- Achieving end-to-end service level management.
- 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 - weather, medical - complex calculations like protein folding, which require processors to run 24/7 for as long as a year at a time.

VII. PROBLEM

- This boom has also led to unprecedented levels of complexity. The simultaneous explosion of information and integration of technology into everyday life has brought on new demands for how people manage and maintain computer systems. Demand is already outpacing supply when it comes to managing complex and even simple computer systems.
- As access to information becomes omnipresent through PC's, hand-held and wireless devices, the stability of current infrastructure, systems, and data is at an increasingly greater risk to suffer outages and general disrepair. We are quickly reaching a threshold moment in the evolution of the industry's views toward computing in general and the associated infrastructure, middleware, and services that maintain them. The increasing system complexity is reaching a level beyond human ability to manage and secure. This increasing complexity with a shortage of skilled I/T professionals points towards an inevitable need to automate many of the functions associated with computing today.

VIII. CHALLENGES

The difficulty in developing and implementing autonomic computing is daunting - enough to constitute a Grand Challenge. At the heart of the matter is the need to bring together minds from multiple technical and scientific disciplines as well as differentiated businesses and institutions to share a sense of urgency and purpose To create autonomic systems, researchers must address key challenges with varying levels of complexity. They are

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

Translating business policy into I/T 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 I/T systems

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

Improving network-monitoring functions to protect security, detect potential threats and achieve a level of decision-making that allows for the redirection of key activities or data. Smarter microprocessors that can detect errors and anticipate failures.

IX. IMMUNE SYSTEM

In a highly connected world, security threats can spread very quickly, and it is vital that the security response be correspondingly quick. One effective response to that threat was the development of a biologically inspired immune system.

The immune system automates every stage of this process. The antivirus software on a protected client system uses a variety of heuristic methods to detect and identify files or other objects that may contain a new virus. A suspect file is encrypted and securely transmitted to an analysis center, where it is exercised and encouraged to spread within a protected environment. After

it spreads, it is automatically analyzed, new detection and repair information is extracted, and the updated definitions are tested and provided to both the original infected system and to any other systems that are registered to receive automatic updates

8.Future

Some 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 etc. In case of SMS, its level of sophistication is Serving the world used for Policy management and Storage tank

(A 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, what level of redundancy, what level of backup, etc. This is goal-oriented management)It's Future goal is Policy language and protocols.

SNMP whose level of sophistication is Heterogeneous components interacting , used for Mounties(enables goal-oriented recovery from system failure instead of procedural oriented recovery), Workload management. It's Future goalisAutonomiccomputingstack,Socialpolicy, Adaptive network routing, network congestion control, high availability clustering have a level of sophistication of homogeneous components interacting.lt is used for Collective intelligence

X. CONCLUSION

Autonomic computing offers as least as many benefits in the security area as it does challenges. The complexity of modern computing systems makes secure systems administration a daunting task and one that is seldom done well in practice. Recent advances, including the growing use of automatic intrusion detection systems, secure embedded processors, proactive security measures, and automated virus response, have helped take some of the burden of security maintenance off overloaded system administrators, but there is much more to do. By making computing systems directly aware of the security policies that apply to them, and giving the systems the ability to conform their actions to those policies, the techniques of autonomic computing will help create systems that are increasingly and consistently secure. 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.

REFERENCES

[1] Jin, Xiaolong; Liu, Jiming (2004), "From Individual Based Modeling to Autonomy Oriented Computation", Agents and Computational Autonomy.

[2] "Autonomic Computing:IBM's Perspective on the State of Information Technology"

[3] 'Trends in technology', survey, Berkeley University of California, USA, March 2002

[4] Kephart, J.O.; Chess, D.M. (2003), "The vision of autonomic computing".

[5] S-Cube Network. "Self-Healing System".

[6] Curry, Edward; Grace, Paul (2008), "Flexible Self-Management Using the Model-View-Controller Pattern".

AUTHORS

First Author – Shama Wankhade, BE 4th year, JDIET Yavatmal, wshama80@gmail.com

Second Author – Rupali Thakre , BE 4th year, JDIET Yavatmal, rupalithakre@gmail.com

Third Author – Payal Koshatwar, BE 4th year ,JDIET Yavatmal, payalkoshatwar@gmail.com