A Simulation of the Fundamental Attributes of Autonomous Computing

Sugandh Shah[1], Hatib Zeeshan[2]

[1]Institute for Development and Research in Banking Technology (IDRBT),
An R&D Centre for Reserve Bank of India, Masab Tank, Hyderabad (A.P), India

[2]Operations Executive, Tata Consultancy Services (TCS), Gorakhpur (U.P), India

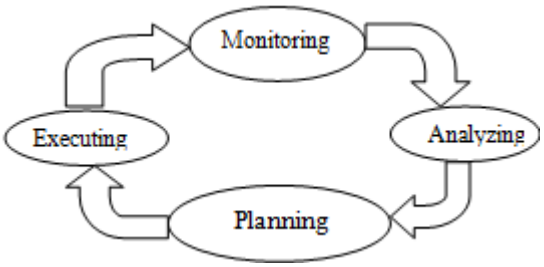
Abstract: ***Autonomous systems are able to adapt to changing environments (such as changes in the resource availability or component failures) in a way that preserves high level operational goals, such as service level objectives. This paper basically focuses on the fundamental attributes of Autonomous computing that are self-healing, self-configuring, self-optimizing and self-protecting. More specifically, the paper presents the detailed design of an Autonomic Download Manager (ADM) which simulates all the fundamental attributes of Autonomous Computing.***

Keywords: Autonomous Computing (AC), AC systems, self-healing, self-configuring, self-optimization, self-protection**.**

1. Introduction

Autonomous Computing (AC) basically refers to the computing infrastructure that adapts (automatically) to meet the demands of the application that are running in it. AC is an emerging paradigm aiming at simplifying the administration of complex computer systems. Efforts required deploying and maintaining complex systems are usually high. Autonomous computing helps to reduce these efforts by allowing administrators to define abstract policies and then enable systems to configure, optimize and maintain themselves according to the specified policies [1].The autonomous systems due to these inherent features are also regarded as intelligent systems.

The general processing strategy of any autonomous system is a cycle of 4 phases. The phases are monitoring, analyzing, planning and executing. The AC system uses this cycle throughout its operational period to maintain its efficiency, safety and high productivity.



Monitoring: In this phase of computation, the autonomous systems scan their work domain and collect information about their current operational scenario and status.

Analyzing: On the basis of data collected in initial phase the system now compares the available information with the performance metrics and predefined scenarios. The outcome of the comparison is a report which contains the current performance statistics of the concerned system.

Planning: With the help of the report generated in 2nd phase, the system plans its further strategies to cope up with the current scenario thereby preparing a set of actions (if any) to be taken in order to improve the performance of the system.

Executing: In this phase the system implements the strategy developed in the 3rd phase by taking the decided actions. The cycle continues and the productivity of the concerned system remains preserved.

2. An Autonomous System: ADM

Here in this context we are about to discuss the design of an autonomic download manager (ADM) as a simulation of the above stated facts of autonomous systems. The ideal processing strategy which must be adapted by every autonomous system can be simulated in the ADM as follows:

Sensor: This unit simulates the monitoring phase in ADM. In this functional unit the ADM keeps track of the available bandwidth and the fluctuating trend possessed by it.

Analyzer: In this unit, the ADM scans the results obtained from the bandwidth monitoring unit and compares it with the predefined metrics to analyze the deviation in the current trend thereby simulating the analyzing phase.

Planner: This unit simulates the planning phase in ADM. on the basis of the deviation detected by analyzer unit, the ADM plans its further strategies to handle the active downloads and cope up with the current scenario.

Effecter: This unit simulates the executing phase in ADM by implementing the performance strategy developed in the above phase, thereby enhancing the overall performance of ADM.

3. Fundamental-X Attributes of Autonomous Systems

The four fundamental attributes of autonomous systems are self-configuring, self-optimizing, self-healing, self-protecting. These fundamentals are often referred as the four self-X paradigms of autonomous systems [1]. According to

these paradigms, system administrators merely specify high level policies, which determine how the system may adjust its behavior at runtime in order to guarantee specified requirements. Administrators are consequently relieved from dealing with numerous details of the system [2].

Self Configuring	Self Optimizing
Self Healing	Self Protecting

Self-configuring:- the autonomous system adapt automatically to dynamically changing environment. When hardware and software systems have the ability to define themselves "on the fly", they are self-configuring [3]. The autonomous systems are capable of configuring and re-configuring themselves under varying and unpredictable conditions.

Self-optimizing:- the autonomous systems monitor and tune resources automatically [3] under self-optimization the autonomous systems efficiently maximize their resource utilization to meet end user needs without human intervention. The system is capable of detecting sub-optimal behaviors and optimizes itself to improve its execution [4].

Self-healing:- the autonomous systems detect and recover from potential problems and continue to function smoothly [4] systems discover, diagnose and react to disruptions. The autonomous systems remains capable of recovering from a failed component by first detecting and isolating the failed component, taking it offline, fixing or isolating the failed component, and re-introducing the fixed or replacement component into service without any apparent application disruption [3].

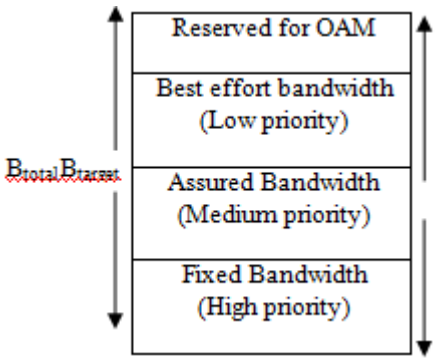
Self-protecting:- systems anticipate, detect, identify and protect themselves from attacks, from anywhere [3]. Self protecting systems have the ability to define and manage user access to all computing resources within the enterprise, to protect against unauthorized resource access, to detect intrusions and report and prevent these activities as they occur, and to provide backup and recovery capabilities that are as secure as the original resource management system.

4. Simulating the four Self-X Attributes in ADM

All the above stated self-X attributes of autonomous systems can be efficiently simulated in ADM. At the time of initiation of downloads the user interface of ADM enables the user to assign priorities to all downloads which are about to be initiated on the basis of their need and importance. After the assignment of priorities when the user commands to initiate the download the URL Connector scans the content about to be downloaded, if the source of the content is found to be specified and the overall content is found to be non-malicious then only the ADM actually continues the downloading, else the operation is terminated. When the content gets downloaded completely the ADM rescans the content of the file to assure of the absence of viruses and other harmful contents, after the scan is complete and no viruses are detected, the ADM now enables the user to access the downloaded content. Hence in such a manner the ADM simulates the *Self-protecting* attribute of autonomous systems within itself.

Now in case, suppose the bandwidth gets lowered beyond the specified limit, in the mid of operation, the ADM at-once suspends the normal execution and resumes with *Priority downloading*. Hence simulates the *Self-healing* attribute of autonomous systems. After switching to priority downloading the ADM starts following the *Dynamic Bandwidth Allocation Algorithm*, [5] the algorithm goes as follows:

Let n be the no. of active downloads that are discovered automatically, B_{total} the total incoming bandwidth, B_{target} the target bandwidth ($B_{target}<B_{total}$), B_{betc} the bandwidth for the best effort traffic class. Bandwidth for operations administration and maintenance (OAM) is reserved and assigned prior to other bandwidths.



Step 1- The high priority grant bandwidth GH_i is assigned for high priority services.

$$GH_i = B_{OAM} + BH_i$$

Where *BHi* is the higher priority provisioned bandwidth for the i_{th} active download.

Step 2- The medium priority services are served before the low priority services. Hence the medium priority grant bandwidth GM_i is assigned as follows:

If$(\sum_{i=1}^{n} RM_i \leq (B_{target} - \sum_{i=1}^{n} GH_i))$
$GM_i = RM_i$
Else if $(\sum_{i=1}^{n} RM_i > (B_{target} - \sum_{i=1}^{n} GH_i))$

$$GM_i = \frac{RMi}{\sum_{i=1}^{n} RMi}(B_{target} - \sum_{i=1}^{n} GH_i)$$

Where $RM_i = QM_i / T_{update}$ is the medium priority request bandwidth for the i_{th} active download and $QM_i$is the medium priority queue length from the i_{th} active download.

Step 3- The low priority grant bandwidth GL_i is assigned for the best effort traffic class. The $B_{betc}$is obtained as

$$B_{betc} = B_{target} - \sum_{i=1}^{n}(GH_i - GM_i)$$

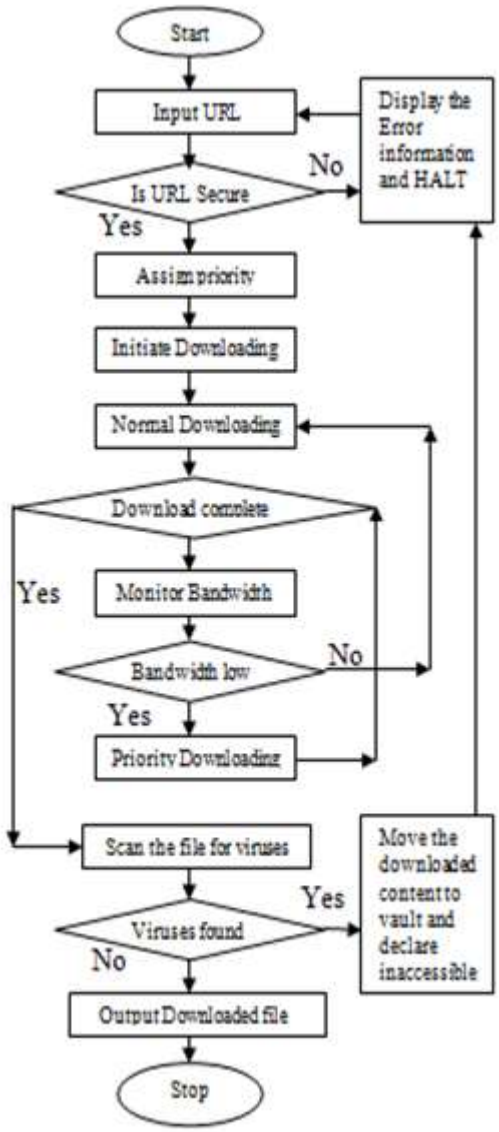
Then GL_i is obtained as

$$GL_i = \frac{RLi}{\sum_{i=1}^{n} RLi}(B_{betc})$$

Where $RL_i = QL_i / T_{update}$ is the low priority request bandwidth for the i_{th} active download, QL_i is the low priority queue length from the i_{th} active download. [5] As soon as the

available bandwidth improves and goes above the specified limit the ADM re-switches to the normal execution strategy (non-priority). Therefore the ADM keeps on optimizing its execution thereby enhancing its productivity and successfully simulates the *Self-optimizing* attribute of autonomous systems.

Hence ADM keeps track of the available resources (bandwidth) and keeps configuring and re-configuring its execution to cope up with the current scenario without affecting its productivity, and hence simulates the *Self-configuring* attribute of autonomous systems. The complete operational strategy of ADM can be summarized with the help of a flow chart as follows:



The overall processing remains abstracted to the end user thereby hiding the complexity of the system.

5. Conclusions

The paper discussed thoroughly the four main self-X aspects of autonomous computing. These features of autonomous systems are expected to solve the service management problems in the areas of software maintenance. The time is right for the full fledge evolution of self-managed autonomous software systems which promise lower total cost of ownership and reduced maintenance burdens. Moreover we also introduced the concept of an Autonomic Download Manager, which is heavily inspired by the aspects of autonomous systems and successfully simulates the fundamental attributes of autonomous computing.

References

[1] Cornel Klein and Reiner Schmid, “A Survey of Context Adaptation in Autonomic Computing”, in IEEE 2008: 4th international conference on autonomic and autonomous system,
[2] J.O. Kephart, “Research challenges of autonomic computing”, in ICSE 2005: 27th international conference on software engineering, New York.
[3] A.G. Ganek and T.A. Corbi, “The dawning of the autonomic computing era”, IBM system journals, VOL 42, NO 1, 2003.
[4] Manish Parashar and Salim Hariri, “autonomic computing: an overview” in 2005:Springer-Verlag Berlin Heidelberg.
[5] Su-il Choi and Jae-doo Huh, “Dynamic Bandwidth Allocation Algorithm for Multimedia services over Ethernet PONs”, in ETRI journal, VOL 24, NO 6, DEC 2002.
[6] H. Muller, L. O’Brian, M. Klein and B. Wood “Autonomic Computing” , April 2006: CMU/SEI-2006-TN-006
[7] J.O. Kephart and D.M. Chess, “The vision of Autoomic Computing”, in IEEE computer society, Jan 2003.

Author Profile

Sugandh Shah is a B. Tech in CSE from Institute of Technology & Management, Gorakhpur. He is pursuing his M. Tech from Hyderabad Central University, and is working on a R&D project from RBI in IDRBT, Hyderabad. His Primary areas of Interest are System Security, Ethical Hacking and VAPT.