ABSTRACT

Information infrastructure is witnessing an evolution with the advent of autonomic computing paradigm. Autonomic Computing Systems (ACS) is becoming more real and visible in present-day computing world, thus creating a context-aware ubiquitous computing environment. Established IT industry leaders have accepted this approach and a great deal of research and development is happening upon this area. Today, systems have grown large accommodating interconnecting complex technologies and practices. This has given rise to the demand of enormous human-skills in low-level-guidance area, engaged in IT support and system maintenance related tasks. Autonomic Systems have the capability of configuring, healing, optimizing and protecting themselves. These Self-managed systems have started becoming part of large enterprises, exhibiting autonomic behavior and tendency for self adaptability. Such systems could be part of a heterogonous – enterprise environment and capable of serving at any level ranging from a simple autonomic component to an enterprise-wide application. Autonomic systems, with the support of more open industry standards can change the way systems behave and respond today.

I. Introduction

Autonomic systems, inspired by the concept of human Autonomic Nervous System (ANS) assist the standard shift from conventional human-manage technology era to technology-manage-technology era. In plain terms, Autonomic systems are those systems that manage themselves with least human intervention. It is an industry wide effort to further progress in the discipline of information technology (IT). Autonomic systems exhibit autonomic behavior by freeing human-intervention from the burden of having to deal with vital but low-level functions [1]. Currently an enterprise-wide information system consists of thousands of networked devices operating under different industry standards exhibiting characteristics with maximum complexity. As more and more devices such as PCs, Notebooks, PDAs, smart-phones become part of the information infrastructure, the complexity factor increases, making it difficult for the most skilled system integrators to install, configure, optimize, maintain and merge systems [2]. This alarming situation will lead to system outages adding more to the cost factor and customer dissatisfaction. Energy firms report a lose of $3 million in revenue per hour, $ 4.5 million per hour for brokerages and $2.6 million per hour for banking firms due to system outage [1][3]. So, a new approach in computing is very much desirable and the answer is Autonomic Computing Systems. New technological paradigms must emerge that effectively integrate these complex technologies, with capabilities to adapt to situations and resolve issues with minimal human intervention. Autonomic Systems must posses self sensing capabilities to accomplish their tasks by anticipating internal or external interactions or disturbances. Systems must anticipate issues or problems, monitor, analyze, plan and execute them by themselves. They must seek resources in case of unavailability and adapt to healthy environments without waiting for human directions or personal intervention. Autonomic systems are never isolated; they are more open than ever and operate purely based on autonomic compatibility characteristics. In an autonomic context a system is defined more from a resource point of view, giving very less preference to human interference.

The growing complexity of computer systems...
emphasizes the need for developing self-managing autonomic systems. The autonomic computing is an evolutionary process and consists of five levels. The first level is called Basic. The further levels are labeled as Managed, Predictive, Adaptive and finally Autonomic. [5,6,7].

Basic: In Basic level an individual manage different tasks and day-to-day operations. It is a starting point in which individuals or IT professionals manage different tasks such as the setting up of the IT environment, monitoring and making updations.

Managed: At Managed level, information is collected from systems through some technologies and tools; that further help to make good and intelligent decisions. This facilitates the system administrator to collect and analyze information more quickly.

Predictive: In Predictive level, predictions and optimal solutions are provided making system more intelligent. Here, new technologies are used which correlate different components of a system to initiate pattern recognition, prediction and suggestions for optimal solutions.

Adaptive: At Adaptive level, information and knowledge is extracted to take different actions automatically. The components use available information and knowledge of the system to take different actions automatically.

Autonomic: In Autonomic level, business policies and objectives are monitored and can change business policies, objective or both. In this last level - Autonomic monitors business policies and objectives; and after analysis this technology has the ability to change the business policies, objective or even both. Through autonomic computing, we achieve minimal human intervention with software or system; maximum use of system resources such as processing power, memory, storage; labor and cost reduction for performing different activities; real-time responsiveness by the autonomic computing environment; and focus by the IT professionals on high value tasks rather than operational tasks.

The four basic characteristics any autonomic systems possess are self configuration, self optimization, self healing and self protecting. These are abbreviated as Self-CHOP characteristics [5, 6]. The metrics or quality factors of these characteristics can be tabulated in table1.

**Table 1: Self-CHOP metrics**

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Self-Configuring</strong></td>
<td>Maintainability, Functionality, Usability, Portability</td>
</tr>
<tr>
<td><strong>Self-Healing</strong></td>
<td>Reliability, Maintainability</td>
</tr>
<tr>
<td><strong>Self-Optimizing</strong></td>
<td>Efficiency, Maintainability, Functionality</td>
</tr>
<tr>
<td><strong>Self-Protecting</strong></td>
<td>Reliability, Functionality</td>
</tr>
</tbody>
</table>

Under self-optimization, the components with respect to Database Management Systems are Query optimization, Automatic Statistics management, Performance monitoring and Intelligence and Parallelism.

1. In Current DBMSs, optimizers (DB2’s LEO, Oracle’s and SQL Server’s Query Optimizer) are used to generate best execution plan against any SQL query or uses existing materialized views automatically [8].

2. Good statistics plays an important role in improving performance and generates optimal Query Execution Plan (QEP) while unnecessary statistics cause performance degradation [9]. Statistics are collected to improve the selection of query execution plan. Current DBMSs provide automatic statistics collection, which includes the number of rows in the table, the number of distinct and most frequent values for a column and the distribution of data values in a column.

3. Current DBMSs provide intelligence by monitoring system statistics, parameters, recent events and applications through different tools like DB2’s Performance Expert, Oracle’s Automatic Database Diagnostic Monitor & SQL Server’s Performance Monitor. These tools improve system and application performance by providing recommendations over bottlenecks and displaying system health.
4. The parallelized feature allows executing multiple activities in parallel. For complex queries automatic parallelism selection can be benefited while in case of simple queries the overhead of parallelism can be bypassed. So in operations that require short duration, parallelism may harm instead of benefit due to switching and communication cost.

In this paper we mainly focus on the various techniques/aspects, theories and current status of Self optimization in autonomic computing.

II. TECHNIQUES AND THEORIES FOR ACHIEVING SELF-OPTIMIZATION IN AUTONOMOUS SYSTEM

Many applications that require continuous performance improvement and resource management make use of control theory approach and active learning based self optimization techniques. Control theory and active learning based techniques have been used to implement self optimization in autonomic systems.

1. Control theory is a theory that deals with influencing the behavior of dynamic systems. Control theory views the self-managing system as a closed control loop that monitors and manages resources. For example, the usage of linear control theory in achieving self optimization has been demonstrated in [11]. In Lotus Notes application [11], performance is evaluated by monitoring the maximum number of users on server. The feedback is provided to an external controller which uses statistical models and previous values to generate an output control. This self optimization technique is evaluated using control theory analysis technique and empirical assessment. A learning based self optimization technique has been used in LEO [12]. LEO implements self-optimization in its query optimizer for DB2. The query optimizer updates the query execution plans based on the feedback generated from early executions. The statistics used for mathematical model of the query execution plan are dynamically updated. It also estimates errors during query execution and re-optimize the query. One approach to achieve this makes use of active learning approach [13]. This approach relies on building statistical predictive models based on the previous history of computing utilities.

2. Adaptive Control Theory involves modifying the control law used by a controller to cope with the fact that the parameters of the system being controlled are slowly time-varying or uncertain. Adaptive control is one of the most dynamic branches in the modern control theory [14]. Adaptive control system based on dynamic object and the environment, by measuring the input/output information, obtains the dynamic characteristics of the managed object and systematic errors in time. According to the change and a certain design method, it makes decisions and modifies the controller parameters autonomously to adapt control signal to meet the changes of object and disturbance, to maintain optimal control performance of system. Feedback control analyses the implementation result of the previous work which is compared with the control standards, for finding bias. And then it identifies the reasons and formulates corrective measures, which can prevent the development or the continued existence of bias. Therefore, through the appropriate expansion, adaptive control theory and feedback control theory can be used to establish autonomic computing system, especially the optimization system. The working principle model of autonomic element based on adaptive control theory is depicted in Fig.1 [15].

![Figure 1: Autonomic element based on adaptive control theory](image_url)
3. **Agent Technology**: Agent called intelligent agent has the properties of reactivity, autonomy and social. It can sense the environment to make a reaction (Reaction Agent) or to achieve the goal-oriented behavior (Deliberative Agent) by plan. Also, agent has been widely known as the key technology which supports the large-scale, open and distributed information systems to achieve dynamic service integration and teamwork. Web Service is a new branch of Web applications, which has the advantages of platform independence, interoperability and so on. Semantic Web technology makes computer understand the meaning of information and complete the intelligent agent function like the human brain. Therefore, based on Agent technology and policy-based management approach, and combined with Web service and Semantic Web technology, we can create a variety of autonomic computing systems. Liao introduced two main methods for building autonomic computing system [22] One is autonomic element based on Reaction Agent, the other based on Deliberative Agent.[29]

4. **Utility Function**: Utility is one of the most commonly used concepts in economics. In autonomic computing, utility function maps each possible state (system performance) of the entity (autonomic element) to a real number, for showing the value which corresponds to system performance (such as reaction time, delay, throughput, etc.). Autonomic computing system uses utility function to show management strategies. Based on the current system performance model, through the maximization of utility, autonomic computing system gets the desired system state and the values of corresponding adjustable system parameters. At last, autonomic computing system adjusts system parameters, making the system achieve the desired state in order to achieve optimization. The autonomic element model of autonomic computing system based on the principle is shown in Fig. 2 [22]. Especially in distributed autonomic computing system, utility function provides a favorable framework for self-optimization. In a dynamic complex environment, utility function can make autonomic element optimize computing resources continuously. Kephart [23] provides a specific example for realization of autonomic computing system. Utility function strategy used to reflect a high-level goal establishes a resource self-management and distribution model for a prototype data center.

There is a strong need for combining with other emerging fields such as pervasive computing and communications for building autonomic systems. Sterritt [28] discusses the combination between soft computing and hard computing to achieve autonomy. Not only the capacity of soft computing which can deal with the imprecision, uncertainty is used, but also the highly predictable solutions and lower computational complexity of hard computing are obtained. Franke [24] proposes the idea that autonomic computing, grid computing and virtualization technology are combined to create autonomic business grid. At present, some researchers have applied model-driven idea to adaptive research [25] and proposed a model-driven self-management approach.

![Utility Function Diagram](image-url)
Another concept is $S^2O_{AC}$ (Service Self Optimization Algorithm). $S^2O_{AC}$ searches the convergence trend of self optimizing function and executes the dynamic self optimization, aiming at minimum the optimization mode rate and maximum the service performance. Based on the integration and updating results of prior self optimization knowledge, $S^2O_{AC}$ includes single dynamic optimization and static optimization prediction. When inner environment changes, the convergence value of optimization function is found out to determine corresponding self optimization mode. When the system inner environment changes, the dynamic self optimization will be executed improving the service performance. When the inner environment is relatively stable, static optimization prediction are implemented [30]. The two procedures interact with each other, to realize the self optimization of system service performance, and the optimization is executed; when system keeps relatively static at a certain time, by the comprehensive analysis of prior optimization knowledge and the updated best optimization mode set, the self optimization prediction model is renewed continuously to optimize the inner resource of system and improve the prediction accuracy.[30]

III. TECHNIQUES FOR PROBLEM DIAGNOSIS AND PROBLEM REPAIR:

Problem Diagnosis involves finding the cause of the problem. Some techniques devised for this purpose are explained below [29]:

Active probing [16] is a problem diagnosis technique that allows probes to be selected and sent on demand. As probes results are received, the system state is updated using probabilistic inference. This process is continued until the problem is diagnosed. It makes use of dependency matrix and Bayesian network to represent probes. The heartbeat failure detection algorithm [17] involves monitored systems sending messages “rising heart beat” to the monitor. Monitor replies by sending “falling heart beat”. Failure is detected when the monitor experiences a delay in receiving of message. The interval of the message is adjusted dynamically, by sending “heart beat period” in the falling heart beat message by the monitor.

Another technique to detect problems in distributed systems is to use tools like Pip [18] that expose structural errors and performance problems by comparing actual system behavior and expected system behavior. Pip provides declarative language to programmers so that they can specify expected system behavior such as timing and resource consumption. It provides tools for logging actual system behavior, visualization and query tools for comparing expected and unexpected behavior. Pip does not rely on statistical techniques to draw inferences and uses explicitly defined expectations to evaluate system behavior.

Performance Query Systems (PQS) [19] can be used to enable user space monitoring of servers. PQS use advanced behavioral models thereby making accurate and fast decisions regarding server and service state. A sensor is placed in the host, which monitors host behavior and publishes significant events to an engine as observations. The engine fuses this information using custom process models and correlates these acquired events with events indicating host-level failure. Output of these process models helps administrators detect the state of nodes. Some problem repair techniques that overcome failures and restore system to normal functioning state are described below:

Component level rebooting is a technique that recovers from defects, without disturbing the rest of the application as shown in Microreboot [20]. However this technique requires that the components must be isolated and stateless. Data recovery is kept separate from process recovery in this technique. The important state information is kept separately in dedicated state stores. This technique provides fast recovery, reduction in functional disruptions and work lost compared to full reboot.
Another safe technique to quickly recover programs from deterministic and non-deterministic bugs is demonstrated in Rx [21]. It rolls back the program to a recent checkpoint upon a software failure, and then re-executes the program in a modified environment.

IV. AUTONOMIC COMPUTING RESEARCH PROBLEMS AND FUTURE DIRECTIONS

At present, some problems of autonomic computing have been studied at National and International level. However, the research of theory, method, and techniques on this new computing model is far from being systematically carried out. There are many problems that have not yet discovered and have to be resolved. Key research problems and future directions are presented as follows:

A. Autonomic Computing System Architecture

Autonomic computing system itself must be the organizational structure, and the combination and interactive collaboration of autonomic element should be limited. However, current research that autonomic computing system should be what kind of architecture to support its elements effectively for self-management and its elements should be what kind of organizational structure to achieve self-configuration, self-healing, self-optimization, self-protection are not yet ripe. Therefore, the research of generic architecture and prototype for autonomic computing system will remain active.

B. Software Engineering Tools for Autonomic Computing System

The development methodology for autonomic computing system is the key point which achieves a blueprint for autonomic computing. We need more powerful software engineering tools for the analysis, design, development, testing and deployment on autonomic computing system. ABLE (Agent Building and Learning Environment) is an agent generation environment based on component reuse. Bigus [26] describes the learning environment ABLE for the development of autonomic computing system. However, it lacks clear structure, agent deployment and operation management environment. The elements themselves also have no clear self-management support. Thus, with the depth research of autonomic computing, software engineering tools will be a research hot spot.

C. Strategies for Autonomic Computing

The research of autonomic computing strategy in theory and in engineering is still in the initial stage. The main problems are as follows: (1) Understanding for the role of strategy in autonomic computing is not deep enough; (2) there is no powerful specification language, as well as strategic planning methods; (3) there is no systematic strategy implement engineering approach; (4) the strategy position on the life cycle of software systems has not been clear. Therefore, some scholars have presented that strategies for autonomic computing will also be focused on [27].

V. CONCLUSIONS

With the promotion of researchers and practitioners, autonomic computing research has infiltrated into pervasive computing, grid computing, software architecture and other fields, it has achieved fruitful research results. However, it remains a relatively immature topic. In this paper, we give an introduction to autonomic computing, techniques and theories for achieving Self-Optimization in Autonomous System, techniques for Problem Diagnosis and Problem Repair and also point out that the current research problems and research prospects hoping to provide a useful reference for the further study of autonomic computing and its self-optimization.

REFERENCES:

3. Parashar Manish and Hariri Salim, Autonomic Computing: An Overview, Rutgers University and University of Arizona, USA.
29. Ruijuan Zheng,Mingchuan Zhang, Qinqtao Wu, Guanfeng Li, Wangyang Wei, “A service Self-Optimization Algorithm based on Autonomic Computing”.
30. Ruijuan Zheng,Mingchuan Zhang, Qinqtao Wu, Guanfeng Li, Wangyang Wei, “A service Self-Optimization Algorithm based on Autonomic Computing”.


Ms. Shanta Rangaswamy, Assistant Professor, Department of CSE, R.V. College of Engineering, Bangalore, is pursuing her PhD from Kuvempu University. Her research areas of interest are Autonomic computing, Data mining, Performance Evaluation of systems, Cryptography and Steganography, System Modeling and Simulation.

Dr. Shobha G., Dean, PG Studies, (CSE & ISE) is associated with R.V. College of Engineering, Bangalore, since 1995. She has received her Masters degree from BITS, Pilani and Ph.D (CSE) from Mangalore University. Her research areas of interest are Database Management Systems, Data mining, Data warehousing, Image Processing and Information and Network Security.