Autonomic Computing for Business Applications

Devasia Kurian
Christ University, Hosur Road,
Bangalore 560029 Karnataka,
India

Pethuru Raj
Senior Consultant Wipro Technologies,
Bangalore 560045 Karnataka,
India

***Abstract*—Autonomic computing, a new deployment technology introduced by IBM a decade ago, to manage the ever increasing complexity of IT systems, has become a part of many large scale deployments today. A lot of inroads have been made by autonomic computing in the areas of networking, data centers, storage, and database management. But few attempts have been exercised in business applications such as ERP, SCM or CRM, and Online Retail. In this paper, we would like to dive deeper to extract and explain where the pioneering autonomic computing paradigm stands today and the varied opportunities and possibilities in this area. A simplistic architecture for deployment of autonomic business applications is introduced and illustrated in this paper. A sample implementation of different management modules from various areas is described in order to invigorate the readers. This should form the basis for a newer and nimbler start, and the ubiquitous application of AC concepts to enable business transformation. This paper represents a solid extension of the paper presented in World Congress on Information and Communication Technologies, 2012[1].**

Keywords—Autonomic Computing Architecture; CRM; Service Oriented Architecture; Multi-Agent Architecture

I. Introduction

Our physical body is intrinsically capable of responding to critical situations, such as an accident or fever in a time-bound manner, to overcome that particular situation. If we are caught in a life or death situation such as face to face with a tiger, the inherent autonomous mechanisms in the body release the right amount of adrenalin, and thereby, keep us alert to face the situation. If any kind of abnormality happens or if any noteworthy changes occur unexpectedly, our highly smart and sophisticated autonomic nervous system anticipates, activates, and adjusts our body functions and parameters. In a second scenario, imagine the activities happening in the body of a person running a marathon. Monitoring of his or her blood pressure, glucose level, body temperature, and sensing of any other body parameters is being carried out by multiple autonomous mechansims. Also, if the body hydration level goes down, appropriate signals have to be dispatched to exhibit the sudden dryness of throat, which in turn prompts the person to drink water. All of these happen autonomously due to the inherent capability that is embedded in our nervous system.

This autonomy characteristic has pervaded every component in our body. Specific tasks are being accomplished by specific body parts, but all these are under the central control of our brain. Correct division and delegation of certain tasks comes handy in achieving the overall autonomy mission. Decision at the top level and execution at the local level is the key differentiator. By doing this task separation in an autonomous way, the central brain can be freed to perform intelligent tasks. Monitoring, management, decision-making, and communication at the central level is the most crucial and critical phenomenon for achieving the desired competencies of the autonomic model.

This art of managing complex situations, by smaller autonomous components and freeing the central system, can be deployed in appliances, networks, services, and applications. They need to be empowered to be self-managing in order to accomplish their assigned functionality and responsibility under all the circumstances. They ought to be equipped to self-diagnose, self-configure, self-heal, self-optimize, and self-protect. With IT autonomy set to grow, the goals of IT agility, availability, and affordability will see a neat and nice realization. The utilization and sharing of various IT resources goes up sharply with the reduction of human intervention, interpretation, and instruction. Ultimately, IT dependability can be guaranteed effortlessly, at a reduced monitoring, maintenance, and management cost.

As IT is getting tightly coupled and synchronized to business goals that are frequently changing, the next-generation IT systems and services need to be produced and deployed as viable and valuable autonomic artifacts so that they behave differently from traditional systems. This helps in moderating the rising multiplicity- and heterogeneity-induced complexity. This is the gist and essence of this new emerging and evolving computing paradigm. Analogous to the animal world, the central brain needs to be freed from mundane tasks. If we take the example of a router in a networking environment, the regular analysis of packet heads and subsequent routing can be taken over by a low level task autonomously. The central system can occupy itself with QoS (Quality of Service) analysis of routes, and change in routes.

The trend of applications governed by business goals is getting accelerated due to numerous and notable advances in in computing, communication, connectivity, collaboration, sensing, perception, and actuation technologies. Right and relevant innovations and inclinations in IT infrastructures and the smart leverage of analytics go a long way in shaping up the business IT.

They help in visualizing all kinds of barriers and to overcome them proactively and preemptively with competent technological improvisations. In this context, the emergence and evolution of autonomic computing is a real boon and blessing for the struggling IT industry. The introduction of autonomic computing in business applications will bring in a disruptive change in this area. In a similar manner as the consistent evolution of natural elements, it will help in the evolution of flexible and futuristic business applications.

II. Survey of Autonomic Journey

In 2001, IBM had internally initiated and investigated the ground-breaking autonomic idea. Through its seminal paper in 2003 [2], IBM had articulated its vision, design principles, development methodologies, and prospects to the outside world. The basic concepts of the MAPE-loop (Monitor-Analyze-Plan-Execute) as fundamental characteristics of an autonomic component and the major self-CHOP (self-configuration, -healing, -optimisation, -protection) characteristics were defined.

As IT systems become more and more complex, there is a need being felt for powerful and cutting-edge technologies and solutions in order to radically minimize and moderate the heterogeneity and multiplicity-induced IT complexity. Professors and professionals have been cooperating to cultivate and inculcate the innovation spirit among students and scholars to bring forth a bevy of competent solutions for the ills afflicting autonomic computing.

Transformational and Optimization Methods - Control systems theory is being leveraged to provide a theoretical and mathematical framework [3] to analyze, in depth, the behavior of autonomic systems and to accurately predict or control parameters such as stability, settling times, and accurate regulation. A variant of differential evolution [4] as the mathematical basis for optimization questions is emerging. Policy frameworks and engines have been established to manage the systems at higher levels of abstraction [5]. Attempts are being made to attain a refinement of policy frameworks using Ontologies [6] and advanced Organic Computing [7]. There are several other theories, techniques, and toolkits to bring in remarkable innovations and improvisations to realize competent autonomic systems of all sizes and scopes.

Domain Penetration – Every kind of electronic systems, especially IT systems, enabling business automation today need well-intended autonomy. Networking and communication fields are prominent and dominant in assimilating and articulating the trailblazing autonomic concepts. A number of research efforts have been initiated and are being sustained to have the brilliant scintillating autonomic features into telecom systems. Network connectivity solutions such as switches, routers, gateways, and proxies are being embedded with autonomy capabilities to behave adaptively.

Robotics is another interesting and inspiring area where the autonomic principles are greatly recognized and rewarded. Humanoids are the latest incarnation of cognitive robotic systems that imitate humans not only in their structures, but also in their behaviors.

Autonomic Solutions - Real-time performance engineering and enhancement (PE2) of mission-critical business systems and server machines can be achieved through the maturity and stability of autonomic concepts. For example, all types of server machines such as web, application, and database systems are being spruced up to ensure high performance and throughput through a host of optimization and automation mechanisms [8]. There are scholarly projects trying to converge the fine and exemplary facets of autonomic computing principles with the fast-growing cloud computing. It is very clear that a number of server and management tasks (user load, workload, job scheduling, provisioning and de-provisioning of various IT resources) are getting automated through software solutions [9].

A few sample projects deployed in the time period starting from 2003 are:

AUTONOMIA [10] presents one of the initial architectures implementing self-configuring and self-healing characteristics. It introduces an Autonomic Middleware service (AMS) comprising of a component and resource repository, and Fault and Security Handlers.

FOCALE [11] introduces a semantically rich architecture for orchestrating the behavior of heterogeneous and distributed computing elements with support of ontologies, policies and knowledge engineering.

PAWS [12] presents an adaptive framework based on web-services. A BPEL (Business Process Execution Language) editor is provided, with which the business process and the constraints in terms of QoS can be defined.

SASSY [13] provides a framework for systems adapting to requirements by selecting services from service providers, that satisfy the utility function. SASSY introduces SAS (Service Activity Schemas), a visual requirements specification language, which describes the required services and activities from the domain ontology, and SSS (Service Sequence Scenarios), which defines the OS requirements. SASSY works in the framework of an SOA architecture to provide the self-architecting framework.

IPAutomata [14] is a part of IPCenter, IPSoft’s commercial product for enterprise wide service delivery, an autonomic component managing multiple intelligent agents.

Current Research Focus – During the past decade, autonomous computing has come out of its infancy, and is slowly becoming a mainstream technology in the areas of networks, data centers, storage, database management and so on. The research focus being discussed currently [15] are interoperability issues in heterogeneous environments, certification, standardization, and so on.

III. Autonomic Computing – Challenges

Despite all efforts and progress made (as listed above) the autonomic computing discipline has still remained a low-key technology and has not gained as much prominence as other computing models such as cloud computing.

The design methodology for autonomic applications has not grown as much as the popular enterprise application frameworks such as the Microsoft .NET framework or the matured Java Enterprise Edition (JEE). The prototype implementations made so far, by various companies across the globe, have miserably failed to ignite widespread technology awareness. Also, the commercial successes achieved of deployed autonomic systems have not raised the pitch for autonomic computing. A thorough understanding of the possible reasons could help us work on these in the future and bring in the deserved importance and desired results to this transformative, disruptive, and innovative technology.

- **Industry Support –** IBM has laid the foundation for the autonomic approach to sharply minimize the complexity of IT systems and has invested a lot in terms of people, money, and so on, in order to take it forward. Unfortunately, IBM has failed to move it into the IT industry in a big way. It has failed to build a larger consortium for formulating standard specifications and to popularize a simplified and streamlined design methodology within the software development community.

- **Less Awareness -** Unfortunately, Autonomic Computing has not been able to penetrate the ecosystem and get wide spread acceptance. It is being promoted as a specialized niche technology. Its visibility and vitality factors are not appropriately disseminated to the IT market.

- **System Characteristics –** The seminal paper from IBM [2] established the self-managing characteristics as fundamental for autonomous systems. Many systems tried to incorporate one or more characteristics rather than focusing on a single aspect. This led to a dilution of resources, which ultimately resulted in weak systems with zero or minimal commercial impact. Furthermore, many of the projects introduced new characteristics like self-management.This resulted in dilution of resources.

- **The Positioning Conundrum –** The ever increasing complexity of IT systems can be solved at three different levels of sophistication as illustrated in the diagram below.

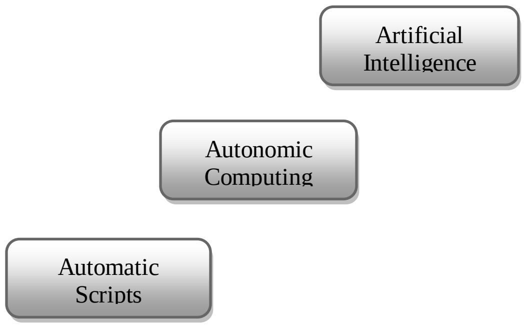

Fig. 1. Autonomic Computing Positioning

Most of the implementers are not clear about this distinction: scripts to carry out certain activities, autonomic systems to manage activities with a feedback loop or artificial intelligence to cognitively solve the issues in the system. During the implementation, they tend to give way to easier scriptsor dig their heads in the myriads of artificial intelligence. The purist view of architecting autonomic computing systems as a simple control system element is forgotten, and the studies go in depth for issues like dynamic location of input services, knowledge transfer, ontologies and so on [16]. This results in researchers spending most of the time fixing semantic or knowledge issues, and thereby forget to exploit the power of the autonomous control loop. Focus should be brought back to the implementation of autonomic computing systems with fixed set of inputs, control function, and control variables.

- **Generic Architecture -** Initial approaches to design autonomous systems came from various corners like control systems, agent-based systems, component-based systems, and so on. The interface mechanism for any agent-based system varied from the interface mechanism for component-based systems. Multiple deviating approaches have destroyed the emergence of a generic architecture and interfaces for autonomous systems.

- **Lack of Standards –** Standards generally inspire worldwide product vendors to produce best-of-the-breed implementations. Standards facilitate interoperability among diverse and distributed system modules. Technologies such as JEE, which have been dominating the technology scenario, are implicitly supported and sustained with simple and pragmatic standards.

- **Lack of Toolkits -** Besides standards, facilitating frameworks, enabling toolkits and platforms, best practices, design patterns, optimized processes, and key guidelines also play a very important role in the massive adoption of technology. Other than ATK (Autonomic Computing Toolkit), a part of the ETTK (Emerging technologies tool kit), produced by IBM, there have not been many simple and sensitive toolkits for AC.

These are few barriers for autonomic computing that responsible for its inability to catch up with its peers. The above issues have to be kept in mind while formulating a framework for business applications. A thorough understanding of the above issues will help one to formulate the right framework for autonomic computing in business applications.

IV. AUTONOMIC BUSINESS APPLICATIONS ARCHITECTURE

Typically, an autonomic system should have multiple independent, yet connected, autonomic components providing a variety of services. The services could be monitoring, providing policy rules, decision-making, controlling, managing, or actuation services. Intelligent software agents are the prime building blocks for autonomic systems and hence they are called multi-agent systems. The individual agents can be components or services implementing a particular function or entities providing multiple services [17]. Policy or rule based engines are the other prominent contributors for systems to be autonomic in their dealings and deliveries. The knowledge manager is a kind of resource manager facilitating insightful access and retrieval of all kinds of information from the knowledge bases, so that appropriate decisions can be contemplated and conveyed.

In short, it is about empowering and embodying software with extra qualities and capabilities. This can be done by employing specially designed engines and knowledge bases so that the software can adaptively exhibit human-like behavior in discharging and delivering their assignments. We have crafted macro-level architecture for an open and standards-based framework for quickly realizing adaptive and self-evolving software. The proposed framework as depicted in Figure 2 will have the following:

1) *Autonomic Managers*
2) *Knowledge Base*
3) *Enterprise Service Bus (ESB)*
4) *Policy Framework and Language*
5) *Reusable assets repository for components (RAR)*

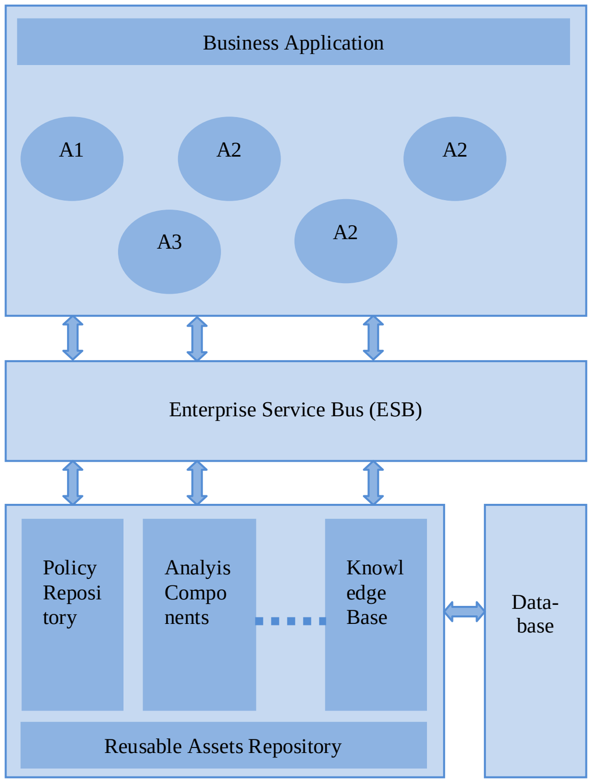

Fig. 2. Autonomic Computing Architecture

The autonomic managers will run as independent processes following a multi-agent architecture. The managers access relevant policies from the repositories and carry out actions based on the specifications in the policies. The evaluation of a policy and its relevant data happens through multiple services provided by various components or agents in the system. To facilitate a dynamic, flexible, and secure access to these services, an ESB framework will be used.

The RAR component will help to maintain and manage components and services in an efficient manner. For small implementations, the ESB and RAR component could be opted out. In such cases, the autonomous managers will directly interact with the component services.

Designing applications as autonomic components and remodeling existing applications as autonomic components are two ways to incorporate autonomy in systems. In the latter case, the vendors should provide service interfaces to monitor or manipulate application data so that third party vendors can create third party autonomic managers. This open approach will help specialists in an area to create relevant autonomic managers, for example a data mining company creating an autonomic manager to find associations using superior associative algorithms, or a text analysis company creating managers to extract key words from data.

Many times cost and time factors are an impediment to creating a system with autonomic managers from scratch. In such cases, existing systems can be extended by autonomic components directly manipulating the database. This is not a recommended practice because there is uncontrolled manipulation of data from multiple sources and therefore, such developments need to be tested thoroughly.

V. Sample Autonomic Computing Applications

The simplicity and application of individual managers is best explained with industry relevant examples. Autonomic managers could be introduced into existing applications, which could control activities of individual modules. It is assumed that the developer of original ERP is not involved in these introductions and therefore, we are dependent on direct database manipulations. If the original developer is the initiator of the introduction of autonomic components, then this could be carried out more elegantly in one of the following ways:

- **Design level:** Autonomic behavior is built in during the design time itself.
- **Interface Level:** At the design level, interfaces are defined to enable other developers to manipulate certain parameters in a controlled manner. This might not be only parameter manipulations, but also interfaces to influence the system status like start, stop, and so on.

The idea of introducing the examples is to:

- Illustrate the options of developing autonomic components for existing applications,
- Initiate the evolution of similar ideas among the domain practitioners and develop standard definitions, interfaces, and so on for independent autonomic component developers.

This would create an ecosystem of core application developers and intelligent peripheral autonomic component developers.

A. Sample Autonomic Computing for CRM

At the outset, CRM is a typical monolithic applicationwhere the introduction of autonomous components might not seem appropriate. In most cases, customers using traditional CRM systems slowly start sinking in the huge volume of data produced by these systems. The introduction of autonomic components into this monolithic architecture brings better manageability and quicker approach to their business goals.

The following sample autonomic computing managers will help the users of CRM systems manage the complexities that are increasing day by day:

Lead Generation Manager: A primary function of any CRM is the management of leads, including all details and communication. The Lead Generation Manager is a self-optimizing autonomic component, which goes into a loop as depicted in Figure 3.

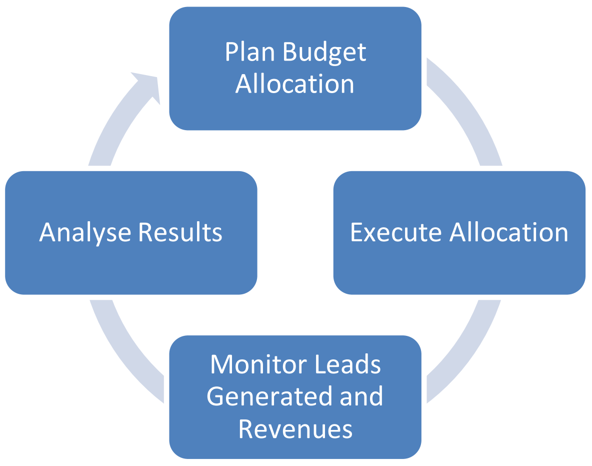

Fig.3. Lead Generation Manager

The administrator specifies in the policy manager that the lead generation should be optimized for a utility function, for example, to maximize the revenue generated by the leads. The optimization could have be carried out based on the number of leads generated, proposals sent, and so on. The budget for lead generation activities is one of the parameters for the policy. The plan component allocates the budget to various lead generation activities – a) online: such as search engines and web-site banners and b) offline: such as exhibitions and print advertisements. The execution takes place automatically, as in the case of many online promotions or manually, as in the case of exhibitions. The revenue generation results are assimilated with pointers to the lead generation activity. This information is analyzed to find out the most effective lead generation media and the budget is reallocated to each of these media based on rules specified in the policy. A simple greedy algorithm might suffice in most of the cases.

PricingManager: A pricing manager could optimize pricing policies across various pricing schemes. The pricing manager may deploy various pricing schemes from season to season and measure the effectiveness of each in terms of revenues, profits, and so on. Based on suggested algorithms, efficient schemes can be selected.

ReminderManager: Customer relationships are maintained by effective reminders of various activities throughout the life-cycle like callback after initial contact, reminder after proposal, activation after a period of inactivity and so on. The optimal results of the reminders differ for each customer segment and product, based on elapsed time, type of reminder, and so on. For example, a customer, who is interested in a buying children health drink might revisit the idea either after 1 week (if she has not yet bought it) or after 6 weeks (to buy again, after finishing the competitor product she bought). The ReminderManager activates various reminders for products, measures the effectiveness, and optimizes the reminder process for the given customer segment and product.

B. *Sample Autonomic Computing for ERP*

Enterprise Resource Planning systems today encompass multiple modules taking care of planning, tracking, and controlling multiple resources and activities in an enterprises. The systems have become so complex, that each enterprise employs a team of engineers to ensure the proper running of the system. This results in huge expenses every year. Often, ERP becomes so complex that it is a mere collection of various modules and interactions between these modules are not regulated. These modules could be production planning, purchasing, inventory, finance, HR, and even extended modules like CRM and so on.

Some of the typical managers, which could be introduced in an existing ERP system are:

InventoryManager: Inventory management is a very important section in ERP, which has great implications on the financial performance of the company. Most of the ERP systems implement simple parameter watches minimum stock quantity, maximum stock quantity, minimum order quantity and so on.

The autonomic Inventory management module collects data from multiple modules such as production, sales, and purchase, executes daily purchase parameters, and measures the effects. The main utility function is the inventory turnover ratio. Sub-parameters such as average inventory cost, and average wait time might also influence the purchase pattern.

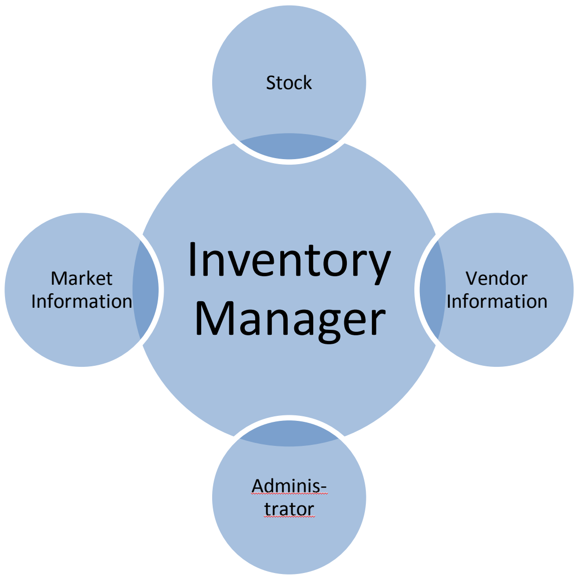

Fig.4. Inventory Manager

Administrator feedback might be sought for special cases, where the autonomic manager might not be able to take a direct decision, for example, when the price of a material drops suddenly. In this case, a lot more information available external to the system is required to decide if it is advisable to procure large quantities for later usage. A query is sent to the administrator giving specifics of the situation and actions are taken based on the recommendation of the administrator.

VendorManager: Long term Vendor Management with selection and rating is very important to the supply side of any organization. Normally, companies keep two or more vendors for each product and divide the purchases, in a manner to optimize the supply, and at the same time keep all of them interested in supplying.

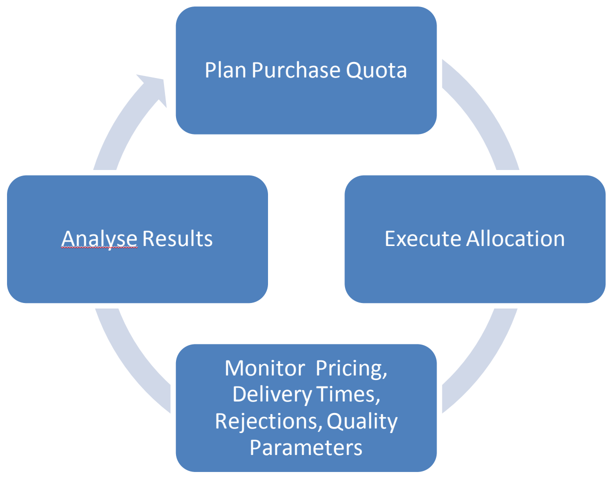
Plan Purchase Quota
Execute Allocation
Monitor Pricing, Delivery Times, Rejections, Quality Parameters
Analyse Results

Fig.5. Vendor Management

The vendors are rated on various feedback parameters such as pricing, delivery times, quality, and so on. Each time a purchase has to be made, the purchase quantities are allocated based on the ratings.

Similar managers can be deployed in multiple areas of ERP such as production, finances, HR, CRM, and SCM. The autonomic integration framework ensures the addition of features in a smooth manner without affecting the existing functionality.

VI. Additional Features

The vision of putting together a software system with the help of multiple autonomic components is similar to the process of putting together hardware components to build a PC. IBM started a similar revolution in the world of desktop PCs by opening up the hardware design, components details, and interface details of the PC. For example, the hard disk or mouse is, to a large extent, autonomous in itself with very well defined interfaces. Multiple vendors came in who specialized in individual components enabling specialization and mass manufacturing, which made the PC prices to plummet. The plummeting prices created an upward spiral for demand and quality of the products. A similar revolution is the long term dream of a software engineer to build such systems.

Some salient features of the above system, other than basic autonomous capabilities, are:

Hot-Plugging: Managers, policies, and services can be changed during runtime of the system, without shutting down the complete system.

Extendable: The simple architecture can extend existing application systems for autonomic capabilities, and can also incorporate advanced features based on AI algorithms, semantic technologies, event processing architecture and so on.

Vendor independent: The autonomic managers could be swapped from one vendor to another, if the performance of one vendor is not satisfactory. The idea is that multiple vendors manufacture autonomic components that compete with each other.

New Technology Integration: Technology is evolving in all field. For example, data mining has brought in a complete new set of intelligence to existing applications. The challenge is to incorporate this new knowledge in existing legacy applications. The independent agent framework introduced by autonomic computing provides an excellent platform to integrate such new intelligence with existing systems.

Module inter-dependencies: Advances in technology bring in new forms of interaction between the various modules in a system. For example, the inventory management in an ERP might be influenced by the sales pattern of another product. This type of cross product influence was not thought of during the conception of the ERP system. Autonomic computing architecture is suited to implement such interactions.

Deployment of autonomic computing architecture is also helpful to achieve the above listed features.

VII. The Evolving Technologies for Autonomic Computing

The simple architecture presented in the previous section can be taken forward with the immense contributions from multi-disciplinary researchers and system professionals. There are many untested yet viable and promising technologies existing and emerging for the reality of autonomic systems. The leading ones include artificial intelligence (AI) technologies for vision, perception, natural interfaces, speech recognition, decision-making and actuation, intelligent agents & multi-agent systems. Technologies such as smart components for software-based solutions, semantic technologies, grid computing, storage, exchange, access, and leverage, service-enablement of IT resources, virtualization, analytical solutions, knowledge engineering, organic computing are also promising.

Services will be empowered through semantic technologies such as ontology in order to be semantic-enabled. Semantic services are dynamic in locating other services and using them to create newer composite services that could meet all kinds of requirement changes. This also brings in intelligent processes.

Process optimization, innovation, integration, integrity, and flexibility along lean processes can be realized through this fast emerging dynamic composition phenomenon. Apart from leveraging semantic services, we will develop and install a taxonomy-based knowledge base. Services capable of reasoning and fulfilling situational (context) changes can readily access the knowledge-base to change their behaviors, to make relevant and rightful structural changes, and to embark on appropriate and preferred decision-making activities in real-time.

Enterprises are extremely event-driven these days. A host of business events, such as goods passing through entrances fitted with RFID readers, are the business realities these days. Enterprise systems need to be equipped with relevant software as well as hardware infrastructures in order to cope with these evolutions, so that they are competitive to their customers, clients, retailers, employees, and other stakeholders. Enterprise infrastructures are therefore being strengthened for receiving, consuming, processing, and routing streams of events at real-time besides triggering timely response and counter measures. There are competent event stream processing (ESP) and complex event processing (CEP) solutions in the marketplace.

With standards-compliant toolkits and SOA workbench, a library of services can be quickly realized, stocked, and leveraged. With the advancements being made in Web 3.0 and ontology engineering domains, constructing semantic services will become relatively easier. In short, to realize adaptive and self-evolving software, we need semantic services for automatic discovery, matchmaking, collaboration, and composition and a knowledge base for self-evolvement.

VIII. The Emergence of Autonomic Cloud Center

With the increasing popularity of autonomic concepts, there will be a seamless convergence and consolidation in the form of innocuously embedding the autonomy characteristics with cloud technologies. The manageability of the ever-increasing complexity of cloud infrastructures, platforms, and software can be simplified only by the evolvement and involvement of higher-level concepts such as autonomic computing.

As the expectations on the cloud technology are on the rise, the focus is on smartly leveraging the autonomic concepts in order to make the cloud more relevant for future IT. As cloud is being touted as the next-generation IT environment for hosting and delivering all kinds of IT infrastructures, platforms, and applications such as services over the open and public Internet communication infrastructure, the relevance of the trend-setting autonomic concepts for the cloud is set to grow exponentially.

Autonomic cloud computing is about empowering cloud infrastructures and platforms to take their own decisions in order to continuously accomplish their assigned tasks. Without any kind of intervention, interpretation, and instruction from humans, cloud systems and services need to consistently provide their functionalities and facilities to subscribers. For example, resource management is one well-known job of cloud IT. As clouds are being positioned as the next-generation IT infrastructure for hosting and delivering a number of applications for personal and professional purposes, the importance of and insistence on bringing cloud-enabled business transformation, simplification, and optimization will only grow.

Cloud computing is leading to transformational changes with stringent requirements on performance or throughput, scalability, security, availability, and extensibility. Their run-time management requires realistic algorithms and techniques for sampling, effective measurement, and characterization for dynamic capacity planning, optimal provisioning, user and load prediction. Models are very important in correctly visualizing the end results and accordingly, all kinds of manipulations can be handled in a better manner. There are load prediction algorithms for cloud platforms. Such algorithms, integrated into the autonomic management framework of a cloud platform, can be used to ensure that the SaaS sessions, virtual desktops, or VM pools are autonomically provisioned on demand in an elastic manner. This approach is suitable to support different load decision systems on cloud platforms with highly variable trends in demand, and is characterized by a moderate computational complexity compatible with run-time decisions.

The value and power of autonomic clouds is rising as a result of the application of autonomic techniques to clouds. This amalgamation results in robust, fault tolerant and easy-to-operate clouds. Such autonomic techniques originate from evolutionary and genetic algorithms, multi-objective, and combinational optimization heuristics, artificial neural networks (ANNs), swarm intelligence and multi-agents systems. These proven and promising techniques can improve the way in which computing systems and applications are built, used, managed, and optimized. Therefore, the benefits for users can be maximized by reducing the operational, maintenance, and usage costs of clouds.

Thus autonomic computing techniques, technologies, and tips are bound to bring in a series of innovations especially state-of-the-art IT infrastructures, platforms, and applications for future IT.

The simple architecture described in the paper also takes care of the deployment of these components in a cloud environment by strictly having multi-agents and SOA for delivering various services.

IX. Future Work

Establishment of a simple architecture for business applications is the requirement of the day to manage the ever increasing complexity of business applications. The industry players should work together to achieve this goal in each domain and create the required definitions, toolkits, and so on to enable multiple vendors to work together in a heterogeneous environment. Some pointers in this direction are:

- **Domain specific Autonomic Definitions:** Vendors of domain specific applications could join hands together to develop specifications of domain control definitions to allow third party vendors to develop autonomic modules. This could be similar to CRM vendors joining hands to define a lead management interface, which could input leads into the system and measure the effectiveness of the leads and thereby, control the lead management campaign. This would lead to a breakdown of a large monolithic application into a collection of smaller components.
- **Dynamic Autonomic Managers:** Vendors can create separate managers that can be integrated into existing systems, such as the Lead Manager for CRM applications. The interface points could be defined at database level or the vendors could create plugins for the popular CRM applications.

- **Analysis Components:** Statistical Analysis or Business Analytics players could provide analysis tools as a service with interface definitions.
- **APIs for Applications:** Application vendors could provide APIs for monitoring and manipulating application specific parameters. For example, a discount interface could allow external applications to manipulate the discount percentages for products available in a system.

Industry forums should come together and actively contribute towards creating definitions of an independent environment to enable the long-term vision of autonomic computing and thereby creating powerful applications. Vendors of specific capabilities such as business analytics should create components and managers in line with the definitions. Similarly, vendors of large applications such as SalesForce, SugarCRM, SAP, and Baan should provide APIs for smaller vendors to interface smoothly with their applications. This ecosystem of all players will push the autonomic computing technology to new heights.

X. CONCLUSIONS

IT systems, solutions, and services are becoming complicated with the addition of more and more features and functionalities. As IT complexity is on the rise, there is an urgent need for easy-to-learn and use complexity-mitigation techniques and tools. The much-discussed and discoursed autonomic computing is being prescribed as the most promising and potential mechanism to significantly slash the unbridled growth of IT complexity. We have listed out the ways and means for taking forward this consolidated and comprehensive method to boost the required IT transformation that in turn will have a telling influence on business agility.

Study of existing implementations has revealed that the deployment of autonomic computing techniques has made some progress in networking, load balancing, power management, and so on, but few inroads have been made into business applications. We have introduced a simplified architecture in this paper, keeping in mind the possibilities of wide spread acceptance from multiple technology corners, and the core trends like policies, knowledge segregation, services, and cloud. The proposed implementation of autonomic agents in the areas of ERP and CRM shows that the technology is well suited to implement complex and sophisticated systems of the future.

REFERENCES

[1] D. Kurian and P. R. Chelliah, "An Autonomic Computing Architecture for Business Applications," in IEEEXplore Digital Library, Trivandrum, 2012.

[2] J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," IEEE Computer Society, pp. 41-50, January 2003.

[3] Y. Diao, L. J. Hellerstein, S. Parekh and Griffith, "A Control Theory Foundation for Self-Managing Computing Systems," IEEE Journal on Selected Areas in Communication, vol. 23, pp. 2213--2221, 2003.

[4] R. Storn and K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," Journal of Global Optmization, vol. 11, pp. 341-359, 1997.

[5] R. Calinescu, "Challenges and Best Practices in Policy-Based Autonomic Architectures," in Third IEEE International Symposium on Dependable, Autonomic and Secure Computing, 2007.

[6] L. Stojanovic, J. Schneider, A. Maedche and S. Libischer, "The role of ontologies in autonomic computing systems," IBM Systems Journal, vol. 43, pp. 598-616, 2004.

[7] H. Schmeck, C. Müller-Schloer, E. Cakar, M. Mnif and U. Richter, "Adaptivity and Self-Organisation in Organic Computing Systems," ACM Transactions on Autonomous and Adaptive Systems, vol. 5, no. 10, pp. 1-32, September 2010.

[8] B. Raza, A. Mateen, T. Hussain and M. M. Awais, "Autonomic Success in Database Management Systems," in ACIS International Conference on Computer and Information Science, 2008.

[9] M. Rak and A. Cuomo, "CHASE: an Autonomic Service Engine for Cloud Environments," in 20th IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2011.

[10] X. Dong, S. Hariri and L. Xue, "AUTONOMIA: An Autonomic Computing Environment," 2003.

[11] J. C. Strassner, N. Agoulmine and E. Lehtihet, "FOCALE – A Novel Autonomic Networking Architecture," 2006.

[12] D. Ardagna, M. Comuzzi and E. Mussi, "PAWS: A Framework for Executing Adaptive Web-Service Processes," 2007.

[13] A. D. Menascé, H. Gomaa, S. Malek and P. J. Sousa, "SASSY: A Framework for Self-Architecting," IEEE Software, pp. 78-85, November 2011.

[14] February 2012. [Online]. Available: http://www.ipsoft.com/.

[15] T. O. Eze, R. J. Anthony, C. Walshaw and A. Soper, "Autonomic Computing in the First Decade: Trends and Direction," in IARIA, St. Marteen, 2012.

[16] R. M. Nami and K. Bertels, "A Survey of Autonomic Computing," Athens, 2007.

[17] L. Zhen and M. Parashar, "Rudder: An agent-based infrastructure for Autonomic Composition of Grid Applications," 2005.