IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 17, Issue 3 Ser. II (May – June 2021), PP 49-56
www.iosrjournals.org

Mathematical Rendition of Generic Process Model-based Design for Decision Making about Cloud Instance Autoscaling Actions

Prasanjit Singh
(M.Sc. (IT), BCA, Department of Computer Science & Engineering, Sikkim Manipal University, India.)
Director – Cloud & DevOps, Starzplay, Dubai, United Arab Emirates.
https://orcid.org/0000-0002-0825-4940

Abstract:
Autoscaling in Cloud Computing is the activity of expanding or diminishing the cloud computational resources like storage, compute, and network services to meet the workload demands. When designed and overseen appropriately, the essential advantage of autoscaling is that the workload (user-requests) gets precisely the cloud computational resources it requires at any random time. A study of auto-scaling architectures, existing techniques, and open issues provides a comprehensive understanding to identify future research solutions. Several methods were proposed for mathematical modeling of the non-stationary process of receiving user's requests, allowing, as a result, forming an analytical model of the workload. The practical significance of the work lies in the possibility of using the developed methods as part of the autoscaling modules of cloud systems, ensuring more efficient management of their infrastructure resources.

Keywords: *Mathematical Models, Cloud Computing, Autoscaling, Process Model-Based Design.*

Date of Submission: 30-05-2021 Date of Acceptance: 13-06-2021

I. Introduction

Due to a substantial growth of the audience of Internet operators rising, and the need for quality of service and users' preference for outsourcing their resources, it becomes critical to forecast application load and procure cloud capacity in real time. Virtualized resources act the same as the physical ones when hosting cloud applications from a cloud. Vertical and horizontal scaling of cloud systems instead of traditional ones is associated with cloud platforms necessitating efficient management. More importantly, this empowers infrastructure to expand as needed and is more responsive to real time changes. Today, distributed computing is thoroughly reforming the manner in which compute resources are designated, making it conceivable to build an autoscaling mechanism on the Cloud. Compute resources can be allocated and on-demand and terminated when not required.

Cloud autoscaling has an elastic behavior. The demand of resources varies according to the ensuing usage. When it comes to cloud autoscaling, there are two established methods namely, scaling-up and scaling-out. Scaling up refers to making a cloud infrastructure component larger and faster so that it can negotiate more load, on the contrary scaling out implies spreading a load out by adding additional resources. Scaling up is also referred to as Vertical Scaling or the method of supplementing the server with additional CPU, memory or network capability.

It should be noted that, scaling out or horizontal scaling, is the process of splitting the workload across multiple compute resources that work simultaneously. Additionally, the requirement of cloud compute resources depends upon certain real-time parameters like:

1. The front-end traffic hits or the number of inbound requests.
2. The back-end load or the number of requests in the server queue waiting to be processed.
3. The duration of time that jobs have been essentially on a 'waiting' status in the server queue.

Based on workload demand patterns, scaling-up and scaling-out allows handling of the request load by appropriately sizing and lining up the resources.

II. Formulation of the Problem

There is no manual that quantitatively or qualitatively delineates how the software organizes to allow for an approach that includes quantitative and qualitative elements. The analysis of eight approaches, conducted by experts for the vast majority of the time, concludes about 95% concordant with the Zachman, IDEF, and SBOK [3]. These three forms are considered to be among the most innovative in terms of architecture and three more advanced ways of providing architecture to industry. Since the ARIS system is only available to members of the company, the corporation's employees, it is an organization dedicated to specific people and purposes. It isn't true that Zachman Business Model 20th century business practices don't take into account current building techniques, which means today's business structures have new characteristics, and many now are very different. To handle the two-way text expansion, prior to this project, no such technique existed. Hence, this project had to be constructed from scratch, as the expansion of text-based hyperlinks (TAF) necessitated the creation of a new methodology. Desires the opportunity of automated resource organization.

As part of a product, the use of this feature would be the use of the automated scaling module in the cloud environment and the service provider of the external method. An automated scale-out approach to distributing public volumes and services provided by cloud providers allows cloud-stored and public data to scale.

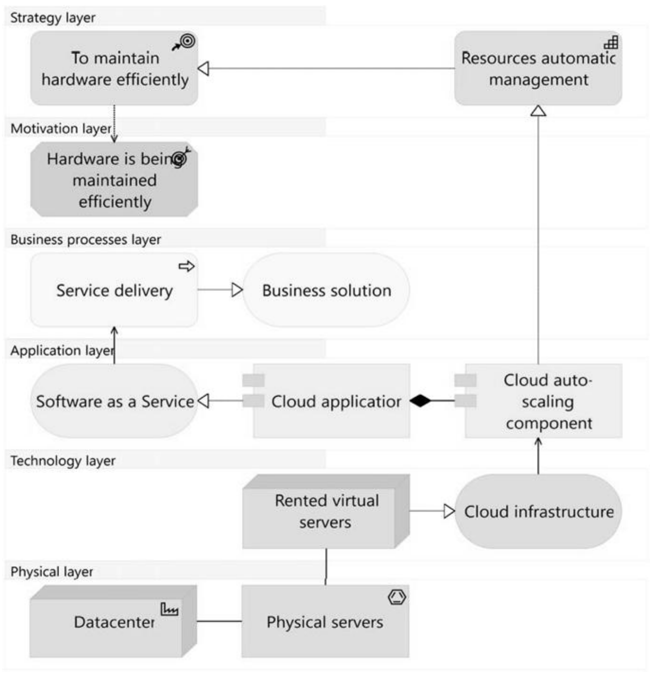

Fig. 1 Layered viewpoint on the autoscaling architecture.

Of the four most commonly used open-source projects, Apache Cloud Stack, Open Nebula, Open Stack and Eucalyptus are investigated, and Open Torus is found to be the most useful [7]. Indirect features (e.g., configurability, security, ability to personalize, and control parameters) were employed, expanding the ability to conduct cloud system analysis with closed-source software. Amongst several new technologies, Amazon AWS and VMware vSphere received particular attention in this latest release. Since all considered systems use instant resource loading as a prerequisite for enhancing the number of running copies of a cloud application, the auto-scaling algorithm also prescribes instant application loading. The load is then decayed from that upper threshold by the next computed instance [when the threshold defined in that one goes above which the next node is started]. When a certain hardware load limit is reached, a similar mechanism occurs- the number of computing nodes decreases by the same sum, as determined by the automated scaling program. In UML notation, the algorithm's activity diagram is shown in Fig. 2.

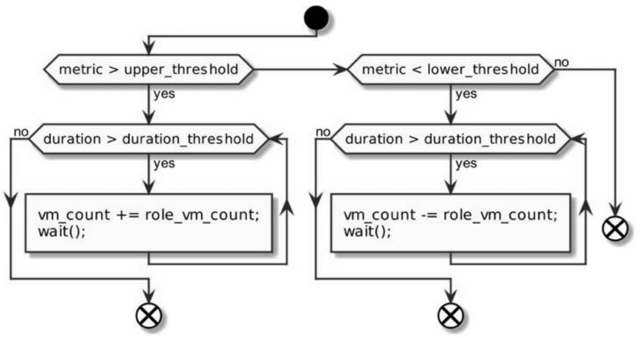
metric > upper_threshold
yes
metric < lower_threshold
no
yes
no
duration > duration_threshold
yes
no
duration > duration_threshold
yes
vm_count += role_vm_count;
wait();
vm_count -= role_vm_count;
wait();

Fig. 2 Activity diagram of native autoscaling algorithm.

In reality, when modeling real-world processes, a variety of assumptions are used. The purpose of such assumptions is to reduce the systems models of which they are a part of their simplest form so that we can quantify their traits while still possessing sufficient models. One of the most important assumptions is that virtual machine processes are stationary. As a result of this assumption, stationary distributions are needed, allowing the use of well-known queuing theory analytical methods.

Real machine operations, on the other hand, do not have the property of stationary. The type of user requests is influenced by a number of factors, including the time of day, the time of year, calendar events, user expectations, and so on. As a consequence, the question arises as to whether the simulation results obtained using stationary distributions are adequate. Furthermore, when developing cloud systems for hosting massive applications, the issue of geographic distribution arises.

Different cloud applications' flow characteristics may include their ability to produce flows in differing periods, strength, such as sporadic, intermittent, or continuous ones. As the server models are not dynamic, nonlinear models are incapable of making estimates about application characteristics that change over time. Require patterns for the same end users cannot be accounted for while servers are in place. Consequently, operational work on non-stationary distributions reflecting variable workloads on cloud systems is unfeasible. A large number of user request flows that could expand the ability to scale an on-demand system as they were built and managed on the cloud architecture were particularly effective in predicting load. Many simulation techniques are utilized to resolve this issue. If the effects of this technique are to be applied to more diverse contexts, it will likely be inadequate; therefore, this technique will only be suitable for those instances where it is very simple to create and understand.

The algorithms used do not take historical data into account and cannot scale based on expected load. This helps us to conclude that the algorithms used under real-world loads are inefficient.

A method for constructing a model of a non-stationary process was proposed because the well-known theoretical methods of mass service theory do not allow for the construction of models of non-stationary processes [10]. The following is a list of non-stationary processes. Periodic, aperiodic (for example, decaying), and chaotic non-stationary are the three forms of non-stationary. The assignment distributions show that. It was decided to investigate the load on network infrastructure in large distributed traffic exchange networks so that adequate models of real-world network activity could be developed [11]. As major traffic exchange points that provide workload information for analysis, M-9 (MSK-IX), OST Dateline, Store Data, LinxTelecom, Trust Info, and Data Space were selected. There are the most sophisticated networks.

Because of the high degree of end-user localization for all considered services, the workload being evaluated thoroughly corresponds to normal user conduct. Similar characteristics can be found in the most common traffic for cloud systems. Such a rule holds for large-scale distributed systems, whose developers usually decompose the workload based on user position when applying the content delivery networks approach. Figure 3 is a graph of network operation obtained from Russia's largest transport network hub [12] "Joint-Stock Company" Center for Interaction of Computer Networks "MSK-IX," from which it can be inferred that the processes occurring in real systems belong to the periodic non-stationary process class.

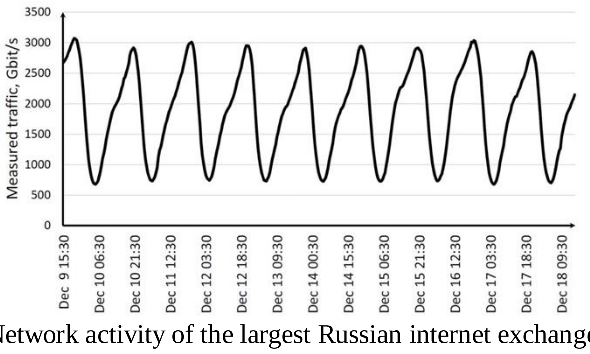

Fig. 3 Network activity of the largest Russian internet exchange point.

The non-stationary distribution (1), which determines the real workload mechanism, is defined analytically in the following form.

$$f(u,v) = a(\gamma_f(v), u)$$
$$\gamma_f(v) = c(v)$$

Following formula (1), any non-stationary distribution can be represented by a set of distribution functions f (x, t) and time dependence of the distribution law parameter $\lambda(t)$. To transition from a real system to a model, a sample of measurements of a given load parameter must first be obtained.

Then, the intensity of incoming requests should be increased. For example, if the calculated value is the time interval between incoming user requests [13], then the instantaneous intensity can be described as the reciprocal. In order to investigate non stationary processes during measurements on a real system, it is important to consider the numerical characteristics of the distribution, which affect the absolute values of the measured, band In order to present the observed process in an empirical manner, it is necessary to automatically estimate the time duration based on a sample of calculated values.

Below, we suggest a method for estimating the time length of such processes in order to ensure their empirical definition and research.

Period Estimation Method

Let there be a periodic non-stationary quantity Y which is time-dependent Y= y(t) with period P, the length of which is to be determined. By tabulating this value over time, one can get a finite series of N values of the value $\{Y_i\}_{i=0}^{N} = y(0), y(1), \ldots\ldots y(N-1)$. Since the task requires low computational complexity and fast convergence [14], simple arithmetic operations were chosen as the basic operations, and the method is a sequence of algorithms described below, as a result of which the value of the period P can be obtained.

Ranking possible period values: If we choose a certain value P ' and perform piecewise averaging of the series $\{Y_i\}$, choosing values with step $P^{\cdot}$, that is form a set $\{Y_{j^{\cdot}}\}_{j=0}^{\frac{N}{P^{\cdot}}} = \{((P^{\cdot} \sum_{j=mp^{\cdot}}^{N/P^{\cdot}} y(j))/N) : m\epsilon\overline{[0, N]}\}$ Then it tries out that $P^{\cdot} = kP,$ where kϵN , the values of the series. Using stationary methods, we can expect to get an estimated value close to the maximum and minimum for the sum by iterating over the results and using the maximum and minimum to set one of the iterated values. If the estimated duration differs significantly from the true one, the outcome of the averaging process will skew toward the line because of the iterated functions' limits. For this estimate, it is necessary for all values of the projected period P to calculate the difference $H_{P^{\cdot}} = \max(\{Y_{P^{\cdot}}\}) - \min(\{Y_{P^{\cdot}}\})$.

Selection of points to evaluate the essence of the feature: At the same time, since the values of the assumed periods can have substantial values, it is necessary to take into account the value of each point in the available sample when calculating $\max(\{Y_{P^{\cdot}}\})$ and $\min(\{Y_{P^{\cdot}}\})$. This method necessitates multiple operations on the main memory and is associated with low and upper bounds of the estimate, respectively productivity. [15, 16] As a result, it is suggested for ranking to take just ten to three points from the averaged feature into account. The number of points used for the assessment is denoted as m. The values of m's boundaries are calculated experimentally. As m<3 is used, the likelihood of a good evaluation approaches 0.5. When m is 10, the primary estimate allows ranking the considered sequence so that the value of $kP^{\cdot}$ is guaranteed to be in the top 10% of the result. The method for determining a particular value of m is shown below.

The choice of the interval between the points was a significant issue resolved in the analysis. Numerous experiments revealed that to reduce the risk of evaluation, the intervals between the considered points should have different values, with the greatest efficiency demonstrated with intervals corresponding to primes (supplemented by the value 1), and scaled to the value of the expected time. For example, if the expected duration is $P^{\cdot}$=27 and you want to analyze the function by m=4 points, you'll need three intervals, which correspond to the intervals between the first four primes: (1, 2, 3, 5). simultaneously, scaling them to the time calculate the step as $S = P^{\cdot} / \sum_{j=1,2,3} j = 4.5$,then growth S by the decremented intermission values.

Selection of estimated period boundary values: Since real-world systems are associated with a high level of random events [17, 18], it is important to optimize the number of piecewise averaged intervals to even out arbitrary emission levels of a function. (Minimize the value of c). At the same time, because of them could be a time or a multi of the original function. If the check fails, the value of D1 and all of its multiples must be put in B, and the loop must proceed to the next iteration.

Otherwise, assume D1 is a multi of the initial function's length and leave the loop. It is important to form the collection of values F, combinations of D1, not found in B, just at the output with the first cycle and Sort the items in this list in ascending order. Then, using the graded set in the loop, perform extensive standard deviation tests, and to make reliable estimates, the data has to cover more closely reflect conditions and control must be given to each weekly cycle to guarantee that a reliable estimates are made once a week. Choosing a value for c2 as the duration of the original sequence allows us to guarantee that at least seven values (corresponding to one day per week) will be included in the potential set for P. Since this is known, we can even draw the conclusion below are the cumulative values. If the tests returned a negative outcome with all values of F, the number D1 is required, and all combinations of that should be put in B, and the process should be repeated one of the first stages of the cycle.

Initial evaluation of estimated periods: After choosing the facts to assess the countryside of the function and the boundary morals for the assessed period, one can generate a key-value bench D based on the foundation series $\{Y_j\}_{j=0}^{M}$, here the estimated period $P'^{\cdot}$ is used as the key. The succeeding step is to produce a set $\{A\}$ of morals that are not purpose of phases. Originally, it is presumed that $\{A\} = \emptyset$.To broad the initial valuation is obligatory in the cycle to achieve the succeeding steps.
1. Eliminate from D all accesses for the keys controlled in $\{A\}$.
2. If D holds no archives, exit the loop. If by this period the estimation of the period did not yield a result, and then whole all designs, resulting that the period could not be strong-minded.
3. Kind D records by the price in plunging order. In case of matching values for altered keys – type by key ethics in rising order.
4. Select the first access 1 *D* as of D and make the detailed typical aberration check labeled below, which indications to the deduction whether the worth 1 *D* can be a era of the unique function or a several of it. If the checked miscarries, the value of 1 *D,* as well as allocates multiples, essential be hired in $\{A\}$ and travel to the succeeding rehearsal of the twist. Then, exit the twist arrogant that 1 *D* is a numerous of the era of the unique function.

At the production of the principal round, it is essential to procedure the set of morals $\{E\}$, multiples of 1 *D*, not enclosed in $\{A\}$ and flourishing this set in rising order. Then, by the hierarchical set in the twist, achieve exhaustive checks on the normal eccentricity and full values, designated below. If for all ideals of $\{E\}$ the assessments gave a undesirable outcome, then the worth 1 *D* is essential and all multiples of it must be positioned in $\{A\}$ and reoccurrence to stage one of the leading round *E.*

Detailed standard deviation check: The check is based on the fact that for the assessed period $P^{\cdot}$, as well as the values of $P^{\cdot}-1$ and $P^{\cdot}+1$, the sum of the typical aberrations of occasionally retelling points is designed as sum$(P^{\cdot})$ sum $(P^{\cdot}-1)$ respectively. Considered set $\{Y_j\}$ is separated into subdivisions in agreement with the patterned passé worth. For example, let the considered set $\{Y_j\}$, j=1, 2, 3, 4, 5, 6, 7, 8, 9 $\{Y_j\}= y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9$. Then, for the estimated period $P^{\cdot} = 3$, the value of $(P^{\cdot})$ *sq* will be calculated as the sum of the standard deviations of the following sets:$\{ y_1, y_4, y_7\}$, $\{y_2 , y_6, y_8\}$ and $\{y_3, y_4, y_9\}$. Founded on the non-stationary possessions, which amplifies sporadic facts, the checked is measured fruitful if *sq(* $P'^{\cdot}$*)* < *sq* $(P'^{\cdot} - 1)$ since for $P'^{\cdot} = kP$ occasionally recapping function standards will have a smaller amount dissimilarity comparative to each further.

It can be realized that in a actual cloud classification additional than one case of a cloud submission often purposes; therefore, the whole torrent of user desires can be disintegrated into discrete drifts for each occurrence of a cloud submission. If we entirety the extreme morals of the concentrations, then we can type the specious deduction that at smallest two bulges need to effort concurrently to ensure the essential presentation [21]. At the similar period, if we gross into version the material on non-stationary, shape logical representations of the load shaped by operators and compute the supreme concentration of the whole movement, excess may not happen when only one bulge is functioning.

Assume there are binary non-stationary deliveries, presented in logical procedure (2) and (3), correspondingly.

$$\begin{cases} f(u,v) = a(\gamma_f(v), u) \\ \gamma_f(v) = c(v) \end{cases} \quad (2)$$

$$\begin{cases} g(u,v) = b(\gamma_g(v), u) \\ \gamma_g(v) = d(v) \end{cases} \quad (3)$$

Process model

By collecting a set of piecewise averages of the intensity function then use a known duration value, one may conduct a piecewise estimating of the intensity function.

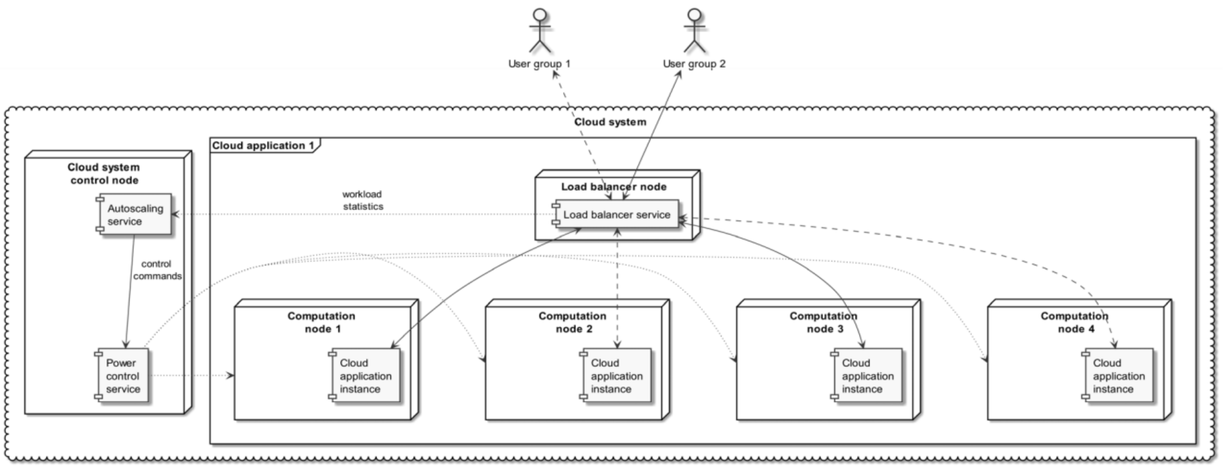

Fig. 4. Process Model of a Cloud Computing System

In these two case studies, the methods for returning the distribution parameters to their original distributions serve the same purpose: They add the values of the function parameters that correspond to each other. The observations were validated via a series of experiments in a simulated environment, in the process of developing the latest version of any Logic Professional. Table 1 is provided as a representation of the results of the experiments is shown below. This employed two non-stationary distributions: 0.03-monotonality to represent the intensity and 1-monophase non-achromatic tungsten carbide and 1-homopolymer about 0.06 seconds each, there was an increment of between 0.1 and 1 in the intensity of the service.

When the intensities' mean values are added together, it is possible to reach the incorrect conclusion [22]. It has been calculated that it will be impossible for a servicing device to cause the entire system to become overwhelmed at 0.09, which means a single servicing device is In this device with an accumulator, the total capacity will decrease because inter-the accumulation loss of power occurs during the cycle. In contrast, interphase loading causes system losses to increase. If you want to know the most intense amount of devices you can use.

III. Results

Table I. Simulation results.

Intensity function	**Phase intensity ratio of incoming waves**	**Maximum load**	**Average utilization**
Sinusoidal	Common mode intensity	1.198	0.853 ±0.004
	The phase shift of the intensity of the second wave by ±	0.901	0.857 ± 0.012
	The phase shift of the intensity of the second wave by π /2	1.108	0.858 ± 0.01
Saw tooth	Common mode intensity	1.199	0.832 ±0.08
	The phase shift of the intensity of the second wave by $\frac{0,03\pm0,06}{2}$	0.998	0.828 ± 0.07

The architecture of the Open Nebula cloud platform has been optimized because of all the previously mentioned points, with the caveat that existing service designs will likely incorporate in an orderly and thoughtful manner as development progresses. Figure 3 shows most of the main architectural elements use by the application. 5. The Single responsibility theory observes when choosing module components. Every module should be responsible for only one module-specific task, and there should be one module that contains all of those components. It serves as the controlling module, as well as implementing, alongside the strategy shown in Fig. of ABC Fig, the remainder of the algorithm is simplified to allow it to run at a lower computational burden.' 2. This module collects and holds the day's regular historical data and then uses it to better estimate the number of nodes needed. An array of various load models stores for each cloud application to predict possible scenarios better. This class contains design and composition models as well as computing resource estimations. The proposed methods are highlighted in working with time series modules consisting of an approximation, interpolation, limit finding, and other common analysis algorithms for working with time series. Time-series functionality is separated from the essential functions and made into a separate feature. It can be more easily expanded to handle machine state needs as needed is provisioning. A component has been developed, which gives users access to functions that enable interacting with the cloud system on demand.

$\sqrt{\quad}$

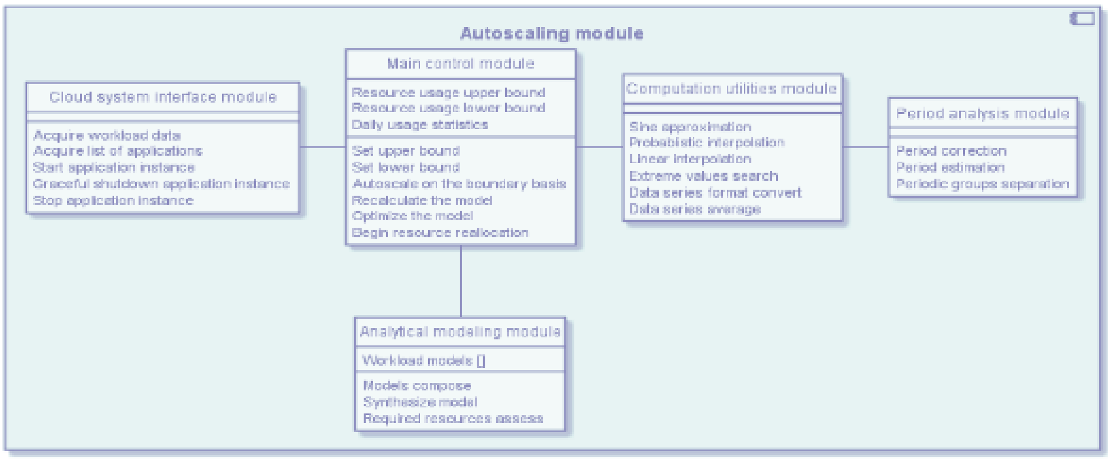
Autoscaling module
Main control module
Resource usage upper bound
Resource usage lower bound
Daily usage statistics
Set upper bound
Set lower bound
Autoscale on the boundary basis
Recalculate the model
Optimize the model
Begin resource reallocation
Cloud system interface module
Acquire workload data
Acquire list of applications
Start application instance
Graceful shutdown application instance
Stop application instance
Computation utilities module
Sine approximation
Probabilistic interpolation
Linear interpolation
Extreme values search
Data series format convert
Data series average
Period analysis module
Period correction
Period estimation
Periodic groups separation
Analytical modeling module
Workload models []
Models compose
Synthesize model
Required resources assess

Fig. 5. Class diagram of the auto scaling software solution

To propose and incorporate all possible methods and modules, Perl v5.18 has been used as the primary programming language. It is capable of being used on both Windows and Linux systems, and is compatible with the Open Nebula cloud platform. The various administrative commands for Open Nebula share identical feature sets (a simple menu for which items can be expanded using the command line, often called "virsh commands"). The solution has Ansible scripts that can make it easier to manage infrastructure deployment and integration in environments that already exist in a couple of places. To the question, Ansible is an open-source software provisioning, configuration management, and application-deployment platform that allows infrastructure as code to be implemented. It runs on a variety of Unix-like systems and can be used to customize both Unix-like and Microsoft Windows systems. It has six functions in the repository for testing, including the ones being used for unit/integration testing. We have used IBM Blade Center HS21 systems with IBM Blade System Storage Blade nodes using the IBM Blade server chassis in the context of a test. To make it easy, we decided to use proxy as a single entry point of contact for user requests in our tests. With our automated installation of Debian a GNU/Linux systems and configuration using Ansible, we have built a set of system images that help our customers install and configure the entire network. The two scripts in the repository perform dynamic proxy reconfiguration. These scripts provided us with the ability to expand our cloud applications dynamically. Just simple CPU code means that just means the applications that run on it use a lot of it. We have run Apache J-Meter load tests on the cloud system, which has found that it is having the proper performance and normal operational behavior. Three test-suite variants are available, each with different workload requirements stored in the repository. Perl has functions used by many open source projects in creating different analytic models and libraries, making the application much more flexible. Perl standard documentation is given in plain text format, so all the code required is supplied to the user. Using the same unit test script is effective for any scenario where you are looking to test periodical information and iterative programming. They allow for the automation of iterative testing. Concerning that, there is an approximate workflow diagram that was created as part of the research.

IV. Conclusion

As a result of the work, and the inferences obtained thereof, a classification function was proposed, sum and transformation expression distributions were combined in representation. Further, a method for quickly estimating the time of a non-stationary process is suggested, which involves successive approximation of the result of calculations to the desired quantity. Several simulation experiments in the Any Logic Professional v7.0.2 simulation environment were used to verify and validate the obtained results. An autoscaling tool for the Open Nebula cloud system was developed to take advantage of existing Open Nebula best practices. Moreover, techniques capable of characterizing non-stationary workloads and better predict cloud application use without needing an overstressed system were proposed. Conclusively, these proposals allow autoscaling systems to function reliably under loaded workload characterization systems and perform dynamic rebalancing of guaranteed-user load models to make analytic models of the cloud flow requests of high precision.

References

[1] Cover, Thomas M., and Joy A. Thomas." Elements of information theory 2nd edition." Willey-Interscience: NJ, 2006.

[2] V. A. Bogatyrev, S. V. Bogatyrev, and I. Yu Golubev. " Optimization and the process of task distribution between computer system clusters." Automatic Control and Computer Sciences 46.3: pp. 103-111, 2012.

[3] Sean Marston, Zhi Li, Subhajyoti Bandyopadhyaya, Juheng Zhanga, and Anand Ghalsasib." Cloud computing - The business perspective." Decision support systems 51.1: pp. 176-189, 2011.

[4] Buzzetto-More, Nicole A., ed. Advanced principles of effective e-learning. Informing Science, 2007.

[5] Lee, Gillam." Cloud Computing: Principles, Systems and Applications/Nick Antonopoulos, Lee Gillam." L.: Springer, 2010.

[6] V.A. Bogatyrev M.S. Vinokurova. " Control and Safety of Operation of Duplicated Computer Systems" Communications in Computer and Information Science, vol. 700, pp. 331-342, 2017.

[7] Jiang, Qiqi, Jianjun Qin, and Lele Kang. " A literature review for open source software studies." International Conference on HCI in Business. Springer, Cham, 2015.

[8] Gogouvitis, Spyridon, et al." Workflow management for soft realtime interactive applications in virtualized environments." Future Generation Computer Systems 28.1: pp. 193-209, 2012.

[9] Mahon, Edward. Transitioning the Enterprise to the Cloud: A Business Approach, Cloudworks Publishing Company, 2015.

[10] T. I. Aliev, M. I. Rebezova and A. A. Russ" Statistical methods for monitoring travel agencies in the settlement system." Automatic Control and Computer Sciences 49.6: pp. 321-327, 2015.

[11] Rafaels, Ray J. Cloud Computing: From Beginning to End. Create Space Independent Publishing Platform, 2015.

[12] V. A. Bogatyrev," Fault tolerance of clusters configurations with direct connection of storage devices." Automatic Control and Computer Sciences 45.6: pp. 330-337, 2011.

[13] Kavis, Michael J. Architecting the cloud: design decisions for cloud computing service models (SaaS, PaaS, and IaaS). John Wiley and Sons, 2014.

[14] Park, Kihong, and Walter Willinger. Self-similar network traffic and performance evaluation. John Wiley and Sons, Inc., 2000.

[15] Rashvand, Habib F., ed. using cross-layer techniques for communication systems. IGI Global, 2012.

[16] Grossman, Robert L." The case for cloud computing." IT professional 11.2: pp. 23- 27, 2009.

[17] V. A. Bogatyrev," Protocols for dynamic distribution of requests through a bus with variable logic ring for reception authority transfer." Automatic control and computer sciences 33.1: pp. 57-63, 1999.

[18] He, Sijin, et al." Elastic application container: A lightweight approach for cloud resource provisioning." Advanced information networking and applications (aina), 2012 ieee 26th international conference on. IEEE, 2012.

[19] Boniface, Michael, et al." Platform-as-a-service architecture for realtime quality of service management in clouds." Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on. IEEE, 2010.

[20] Mao Ming, and Marty Humphrey." A performance study on the vm startup time in the cloud." 2012 IEEE Fifth International Conference on Cloud Computing. IEEE, 2012.

[21] V. A. Bogatyrev and A. V. Bogatyrev. " Functional reliability of a realtime redundant computational process in cluster architecture systems." Automatic Control and Computer Sciences 49.1: pp. 46-56, 2015.

[22] Chhibber, Anju, and Sunil Batra." Security analysis of cloud computing." International Journal of Advanced Research in Engineering and Applied Sciences 2.3: pp. 2278-6252, 2013.

Prasanjit Singh, et. al. "Mathematical Rendition of Generic Process Model-based Design for Decision Making about Cloud Instance Autoscaling Actions." *IOSR Journal of Mathematics (IOSR-JM)*, 17(3), (2021): pp. 49-56.