Scady: A Scalable & Dynamic Toolkit for Enhanced Performance in Grid Computing

Rakesh Bhatnagar
MCA Department, S. K. Patel Institute of Management & Computer Studies
Kadi Sarva Vishwavidyalaya University
Gandhinagar, Gujarat 382023, India
rakesh_bhatnagar@rediffmail.com

Dr. Jayesh Patel
MCA Department, Acharya Motibhai Patel Institute of Computer Studies
Ganpat University
Kherva, Gujarat 382711, India
jayeshpatel_mca@yahoo.com

Abstract - Grid computing has resulted in faster execution of applications. Day by day the requirement for execution speed of applications is increasing. To achieve this, suitable middleware is needed and to be configured requirement specific. An application can be executed faster only when there is no node failure or network failure. Moreover, at run time, if a node fails, there should be some alternative arrangement to reconfigure other node to complete the task which was supposed to be done by the failed node. Thus, a middleware is needed that can be configured according to the requirements of the application. Such a middleware would help in increasing the performance of the application. In most of the middleware available presently, dynamic configuration is not facilitated. This has become one of the reasons for the failure of adoption or implementation of Grid in many organizations.

This paper analyzes the present GUI Grid – Alchemi with its components and challenges. It also proposes a toolkit-Scady which solves the challenges of Alchemi and provides higher performance. Two experiments are done and their result analysis shows the performance of the proposed toolkit.

Keywords: Managers, Executors, Node failure, Session, Reallocation

I. INTRODUCTION

High performance computing is the need of the hour. Organizations and enterprises now want their applications to execute as fast as possible and without any faults. The demand for the use of distributed resources and computing cycles are increasing day by day. Moreover, the power present in the networks (LANs) is not being utilized properly. To achieve the need of the hour, Grid computing has provided the solution. There are several Grid systems developed till date to achieve high performance computing. Some of them are success, but many of them failed due to various issues like dynamic node failure [1][6][7], security [2] etc.

Still grids are the most common alternative for performing distributive computing and utilizing the distributed resources. Unix Platform is used primarily to implement Grids. This severely limits effective utilization of the computing resources of desktop computers running on Windows platform. Companies and organizations prefer windows based i.e. GUI based solutions for targeting high performance computing. Grid Computing is a special type of high performance computing that relies on distributed resources like CPU cycles, Storage, Power supply, Bandwidth etc. The idea is to perform the work of Super computer by using all these resources together.

One such Grid toolkit that is Windows based and is becoming popular in the grid community is Alchemi [9][10]. It works on .NET platform and provides the features supported by it like Remoting [5][6] and Web Services [7]. Another toolkit named Scady[1] which we have developed based on Alchemi is compared with the Alchemi Toolkit itself. The results obtained by Scady are much better than Alchemi Toolkit as the features that are not supported in Alchemi are provided in this toolkit. The API of Alchemi is studied and then extended in Scady to provide all those features.

To show the efficiency of Scady toolkit, a factorial program for finding the factorial of a number is run by using the Alchemi and Scady toolkit. Next a program for calculating 2^{32} (2 raise to 32) is executed using both the toolkits. Scady computes the value of both the programs in less time as compared to Alchemi toolkit.

Paper is organized as follows –Alchemi toolkit is discussed in section 2. After Literature review, the challenges that are present are identified. Based on this a new toolkit is developed. Section 3 highlights the working of the proposed toolkit. Section 4 contains the case studies, Section 5 is the conclusion, section 6 describes the future work and section 7 contains the references.

II. ALCHEMI TOOLKIT

Alchemi [9][10] is a GUI grid toolkit that is developed in .Net to deal with key issues of security, reliability, scheduling, and resource management. Microsoft .Net platform supports all of these features. It actually provides features such as remote execution and development of web services, multithreading, disconnected data access and managed execution.

The key features supported by Alchemi as listed in [10] are:

- Without a shared file system, clustering of desktop computers based on internet;
- Hierarchical and cooperative grids through federation of clusters;
- dedicated or non-dedicated execution by nodes;

- Grid thread programming model is object oriented and
- Cross-platform interoperability

There are four components in Alchemi .NET grid - User or Owner, Executor, Manager and Cross Platform Manager [10].

WORKING –
User (an interface) requests for a task to be done. This task is given to the Manager. Manager is generally installed on the server. However it can also be installed on a client machine. Manager then delegates the task to the executors which are running on different client machines or nodes. Executor executes the task and sends the output to the Manager which then stores it in database and gives it to the User GUI.

CHALLENGES –
Following challenges are found in Alchemi -
- If an Executor i.e. a node fails due to any reason, the task is stopped and the computing remains unfinished.
- If the Manager fails due to any reason, the whole processing is stopped.
- Inter thread communication is not supported.
- There is no peer-to-peer communication between Managers that can solve the problem of node failure.
- There is no policy for allocating resources.
- OGSI standards are not completely followed.

III. SCADY TOOLKIT

Scady (Scalable & Dynamic) is a windows based toolkit developed for Grid systems. It is developed with the objective to provide the features such as multithreading, inter-thread communication, task reallocation in case of node failure etc. It is developed based on Alchemi toolkit. API of different grids are studied and are extended to provide the solution of the problems listed in above section. There are two components of Scady – A Manager and an Executor.

WORKING –
A Manager is installed on each cluster or the location. We can also install more than one Manager at one location (through inter thread communication model). This will help in removing the problem of failure of any one Manager. This supports the VPN concept. Each Manager can have multiple Executors under it.

When any task is to be done, Manager delegates the task to Executors. The result is processed by the executor and sent to the Manager. This working is shown in Fig – I.

In Manager, IP address and Mac Address will be displayed reading from the system. All the connected nodes in the network will be displayed. Nodes will be selected through check box and connection is established. Managers will take less time to produce result if the numbers of executors are more.

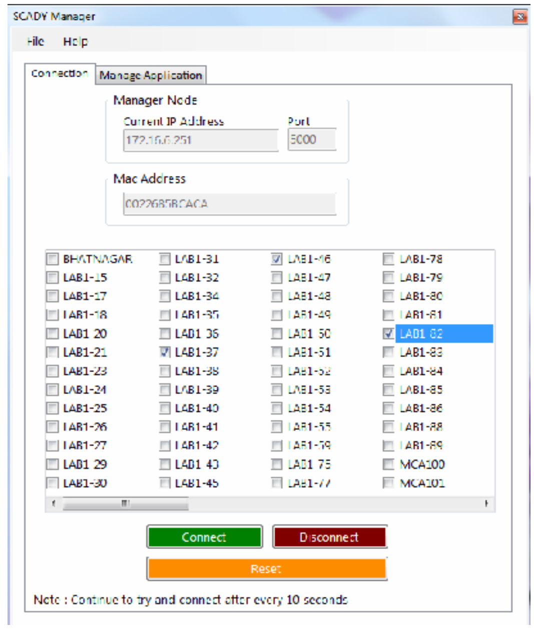

FIG-I: SCADY MANAGER (CONNECTION)

If due to any reason an executor goes down in between the task, Manager dynamically reallocates it to another executor and the task is completed. For this the Manager has to dynamically monitor the executors. This is implemented by checking the connection every 10 seconds through the session variable. If a node fails, the task is reallocated to another node by manager based on the CPU load of nodes. After the connection is done, application is selected. Presently four applications have been taken. These applications can be increased with the use of database. Input values will be provided and then application is started. This application is processed by the executors and the result is sent to the manager. This working is shown in Fig – II

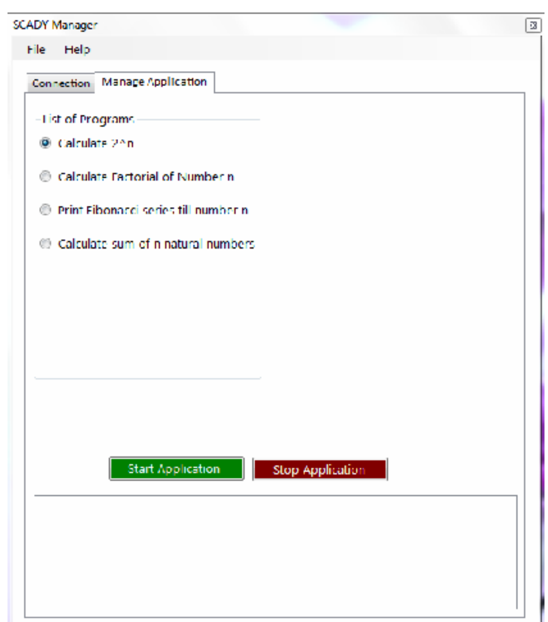

FIG-II: SCADY MANAGER (APPLICATION MANAGER)

With this model, node failure and unfinished task problem is solved. Also as the number of managers and executors are more, the time taken to complete the task is less as compared to Alchemi .Net toolkit. For this we performed two experiments which are described in the next section.

IV. CASE STUDY

EXPERIMENT - I

Experiment is done to write a program to find factorial of a number. The network taken for the experiment is intranet (campus network) in which there are 4 sub-networks, each sub-network having a minimum of 50 client machines. Total computers in the network are around 300. Hardware and Software configuration used are - nodes with 3 GB RAM and 4 GB RAM, 2.66 GHz, Core-to-Dual Processors, Core-to-Quad Processors and i3 Processors, 24 ports Network Switch, Visual Studio 2012, .Net Framework 4.5 and SQL Server Express Edition. The test was done with 3 nodes (Executors), then with 10 and then 50 nodes. The results are shown in Table-I.

TABLE-I: FACTORIAL PROGRAM

Number of Executors	Time (in Secs)	
	Alchemi Toolkit	Scady Toolkit
3	0.81	0.46
10	0.62	0.39
50	0.59	0.25

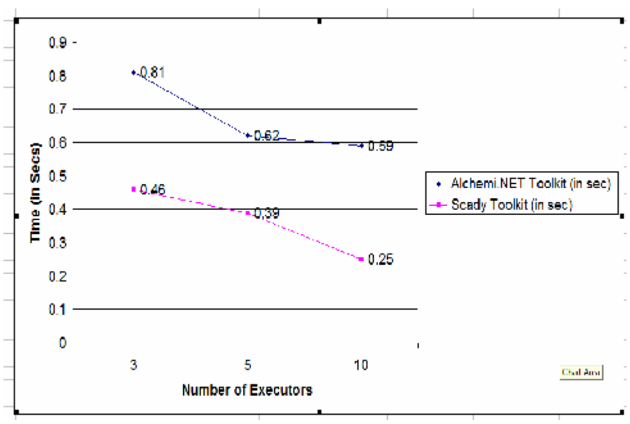

FIG-III: EXECUTION TIME OF FACTORIAL PROGRAM

ANALYSIS

The results of the experiment of calculating factorial of a number by using 3 executors, 10 executors and 50 executors are shown in Fig-III. It can be noted that the time taken to calculate factorial of a number decreases when the number of nodes increases. Also time taken by Scady toolkit is less.

EXPERIMENT – II

Another experiment was done for calculating 2 raise to 32 i.e. 2^{32}. The test was done with the same hardware & software configuration and selecting 3, 10 and 50 nodes. The working of Scady Manager is shown in following figures.

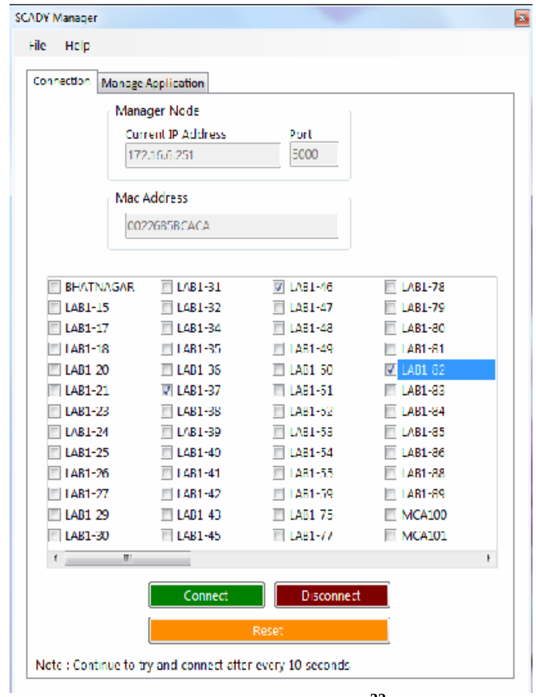

FIG-IV: SCADY MANAGER (FOR 2^{32} PROGRAM)

3 nodes are selected from the network – Lab-37, Lab-46 and Lab-82. After connection, application is selected and started.

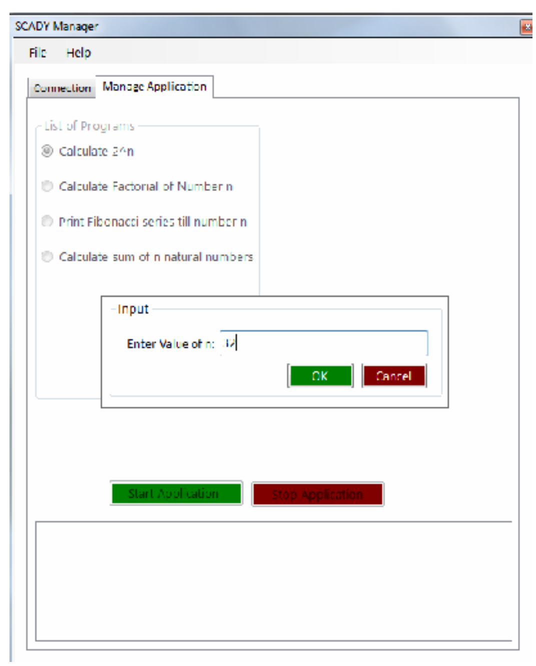

FIG-V: SCADY MANAGER (FOR 2^{32} PROGRAM)

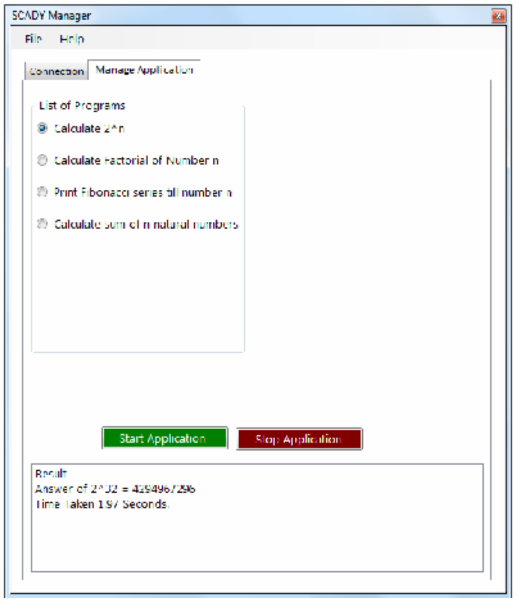

FIG-VI: SCADY MANAGER (FOR 2^{32} PROGRAM)

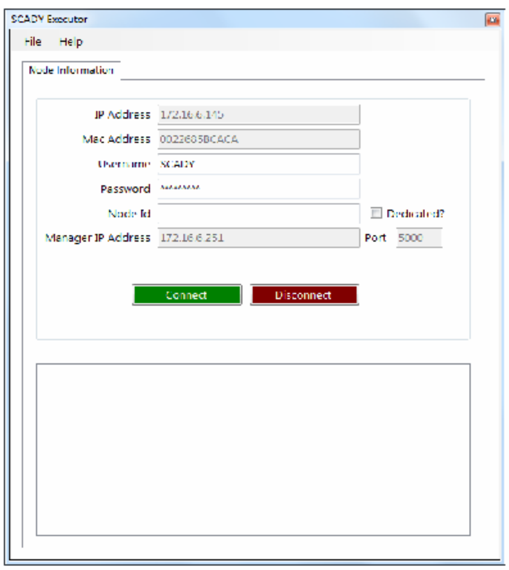

FIG-VII: SCADY EXECUTOR

The result and time taken for calculation is printed in the result window. This experiment is repeated for 10 and 50 nodes also. The results are given in Table-II.

TABLE-II: 2^{32} PROGRAM

Number of Executors	Time (in Secs)	
	Alchemi Toolkit	Scady Toolkit
3	2.62	1.97
10	2.34	1.65
50	1.81	1.23

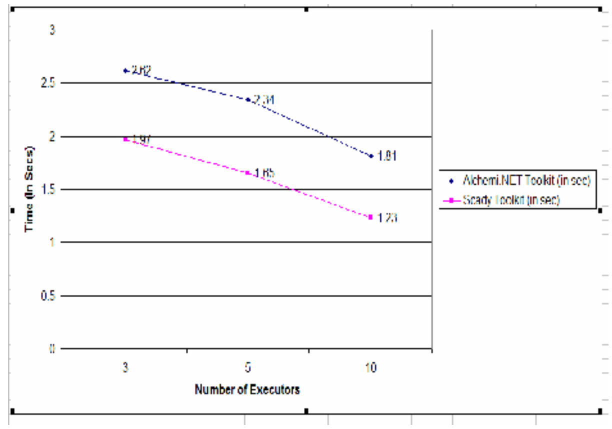

FIG-VIII: EXECUTION TIME OF 2^{32} PROGRAM

ANALYSIS

The results of the experiment of calculating 2^{32} by using 3 executors, 10 executors and 50 executors are shown in Fig-VIII. It can be noted that the time taken to calculate 2^{32} decreases when the number of nodes increases. Also time taken by Scady toolkit is less.

V. CONCLUSION

In this paper, we studied Alchemi .Net toolkit for Grid Computing & its APIs. We found that even though, Alchemi .Net succeeded in providing sufficient performance in grid computations, it is not easy to provide a dynamically configurable environment that can help in ensuring that the computation does not fail even if any node or executor fails. To solve this problem, we have constructed a small grid toolkit – SCADY and tried to add and extend the existing API of the available middleware. The connection to node is checked every 10 seconds and if a node is found to be failed, the task is allocated to the next available node with less CPU load.

We conducted two case studies on both the toolkits and found that Scady toolkit provides the output in less time as compared to Alchemi. Also the time taken is very less when the number of executors is increased.

VI. FUTURE WORK

The proposed toolkit has opened up a window to wider applications. We plan to extend the Scady toolkit to even more complex applications and support of CPU load based resource allocation. Also, instead of checking the status of node every 10 seconds, we are planning some mechanism that can be used through which the status for failure of task or node is reported to the manager immediately. (Very much like ICMP message delivery).

VII. REFERENCES

[1] Rakesh Bhatnagar, Dr. Jayesh Patel, "API Specification for Small Grid Middleware - Scady", 11th IEEE India Conference INDICON 2014 Emerging Trends and Innovation in Technology, Dec 11-13, 2014, Pune.

[2] Rakesh Bhatnagar, Dr. Jayesh Patel, "An Empirical study of Security issues in Grid Middleware", International Journal of Emerging Technology & Advanced Engineering , ISSN 2250–2459**,** Volume 4 Issue 1, Jan 2014, Pages 470-474

[3] Rakesh Bhatnagar, Dr. Jayesh Patel, "Performance Analysis of a Grid Monitoring System - Ganglia", International Journal of Emerging Technology & Advanced Engineering, ISSN 2250–2459**,** Volume 3 Issue 8, Aug 2013, Pages 362-365

[4] Rakesh Bhatnagar, Dr. Jayesh Patel, "Performance Analysis of a Grid Monitoring System - Autopilot", Journal of Sci-Tech Research (KSV – JSTR) - Volume 4, Issue 2, July - Dec 2013, ISSN 0974 – 9780, Pages 15 – 20.

[5] M. Venkateswara Reddy, A. Vijay Srinivas, Tarun Gopinath, and D. Janakiram, "Vishwa: A reconfigurable P2P middleware for Grid Computations", http://dos.iitm.ac.in/LabPapers/icppVishwa.pdf

[6] Hirunavukkarasu Sivaharan, Gordon Blair, and Geoff Coulson, "GREEN: A Configurable and Re-configurable Publish-Subscribe Middleware for Pervasive Computing", On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, Lecture Notes in Computer Science Volume 3760, Springer 2005, pp 732-749

[7] Kaoutar El Maghraouia, Travis Desella, Boleslaw K. Szymanskia, James D. Terescob, and Carlos A. Varela, "Towards a Middleware Framework for Dynamically Reconfigurable Scientific Computing", Elsevier, Advances in Parallel Computing, Volume 14, 2005, Pages 275–301 and http://www.cs.rpi.edu/~szymansk/papers/hpc.04.pdf

[8] Manisha Bhardwaj, Sarbjeet Singh & Makhan Singh, "Implementation of Single Sign-on and Delegation mechanism in Alchemi.Net based Grid Computing Framework", International Journal of Information Technology and Knowledge Management January-June 2011, Volume 4, No. 1, pp. 289-292

[9] Rajkumar Buyya, Akshay Luther, Rajiv Ranjan, and Srikumar Venugopal, "Alchemi: A .NET-based Enterprise Grid Computing System", Internet Computing, 2005 www.cloudbus.org/papers/alchemi_icomp05.pdf

[10] Rajkumar Buyya, Akshay Luther, Rajiv Ranjan, and Srikumar Venugopal, "Alchemi: A .NET-based Grid Computing Framework and its integration into Global Grids", www.cloudbus.org/papers/alchemi.pdf

[11] Paul Grace, http://www.academia.edu/6142596/A_component based_middleware_framework_for_configurable_and_reconfigurable_Grid_computing

[12] Yuhui Deng, Frank Wang, Na Helian, Sining Wu, Chenhan Liao, "Dynamic and scalable storage management architecture for Grid Oriented Storage devices", Elsevier, Parallel Computing, Volume 34, Issue 1, January 2008, Pages 17–31

[13] T. Purusothaman, S. Annadurai, H. Vijay Ganesh, C.T. Chockalingam and B. Uthra Kumar, "Dynamically Scalable, Heterogeneous and Generic Architecture for a Grid of Workstations", Journal of Grid Computing (2004) 2: 239–24, Springer 2005

[14] Kaoutar El Maghraouia, Boleslaw K. Szymanski, and Carlos A. Varela, "An Architecture for Reconfigurable Iterative MPI Applications in Dynamic Environments", Parallel Processing and Applied Mathematics, Lecture Notes in Computer Science Volume 3911, 2006, pp 258-271 http://link.springer.com/chapter/10.1007/11752578_32

[15] R. Nagaraja, Dr. G. T. Raju, "COMMPC - Component Based Middleware for Pervasive Computing", IJCSNS International Journal of Computer Science and Network Security, VOL.11 No.9, September 2011

[16] Gianpaolo Cugola, Gian Pietro Picco, "REDS: a reconfigurable dispatching system", SEM '06 Proceedings of the 6th international workshop on Software engineering and middleware, Pages 9 – 16

[17] Kaoutar El Maghraoui, Travis J. Desell, Boleslaw K. Szymanski, and Carlos A. Varela, "The Internet Operating System: Middleware for Adaptive Distributed Computing", International Journal of High Performance Computing Applications, Vol. 20, No. 4, Winter 2006, pp. 467-480

[18] Carlos A. Varela, Paolo Ciancarini, Kenjiro Taura, "Worldwide computing: Adaptive middleware and programming technology for dynamic Grid environments", Journal Scientific Programming - Dynamic Grids and Worldwide Computing, Volume 13 Issue 4, October 2005, Pages 255-263

[19] Coulson, G., Blair, G.S., Clark, M., Parlavantzas, N., "The Design of a Highly Configurable and Reconfigurable Middleware Platform", ACM Distributed Computing Journal, Vol 15, No 2, pp 109-126, April 2002.

[20] Kon, F., Costa, F., Blair, G.S., Campbell, R., "The Case for Reflective Middleware: Building Middleware that is Flexible, Reconfigurable, and yet simple to Use", CACM, Vol. 45, No. 6, pp 33-38, 2002.

[21] Geoffrey Fox, Shrideep Pallickara, and Xi Rao. A scaleable event infrastructure for peer to peer grids. In JGI '02: Proceedings of the 2002 joint ACM-ISCOPE conference on Java Grande, pages 66.75, New York, NY, USA, 2002. ACM Press.

[22] A. Carzaniga, D.S. Rosenblum, and A.L. Wolf "Achieving Expressiveness and Scalability in an Internet-Scale Event Notification Service", 19th ACM Symposium on Principles of Distributed Computing (PODC2000), Portland OR. July, 2000