International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
NCACI-2015 Conference Proceedings

using of the Radar Charts in the Software Frame work of the Self Adaptive Systems

C. Govardhan M.Tech
Asst.Professor
Department of CSE
KMMITS, Tirupati

C.C. Kalyan Srinivas M.Tech
Asst.Professor
Department of CSE
KMMITS, Tirupati

***Abstract*-In the recent years of the distributed computing, many trends have been emerged for the benchmarking of the self adaptive systems. Even though architecture of the software framework for evaluating and benchmarking of the self adaptive distributed system proposed a meaningful way for understanding the benchmarks. It will not be given the precise view of the different benchmarks, In this paper , using of the radar charts in the framework of software component was proposed ,which gives the coherent view of the different metrics.**

I. INTRODUCTION

Traditionally, handling changing requirements, faults, or upgrades on different kinds of software-based systems have been tasks performed as a maintenance activity conducted by human operators at design or development time. However, factors such as uncertainty in the operational environment, resource variability, or the critical nature of some systems which cannot be halted in order to be changed, have lead to the development of systems able to reconfigure their structure and behaviour at run time in order to improve their operation without any human intervention. This kind of systems, which typically operate using an explicit representation of their structure and goals, has been studied within different research areas of software engineering (e.g., component-based development, requirements engineering, software architectures, etc.) and described with different names, which put their emphasis on different aspects. From those different names (self-healing, self-managed systems, etc.)

Those are called Self Adaptive sytems **.**Metrics are always domain specific which evaluates the performance of the self adaptive systems and it is always problem specific ,as such it is hard or even impossible to mention a metric which is useful to every domain.In this paper , a radar chart was proposed in the context of the software systems .

II. RELATED WORK

[1]Describes the use of Declarative Benchmarking Definition Language (DBDL), and Architecture of the software framework for evaluating and benchmarking self-adaptive distributed systems. Regarding the radar charts and its advantages was mentioned in [2]. In this paper we will be using Radarcharts for clear explanation of different domain specific metrics.

III. RADAR CHARTS

Radar charts also called spider charts, polar charts, or kiviat charts are a form of a graph that allows a visual comparison between several quantitative or qualitative aspects of a situation, or when charts are drawn for several situations using the same axes, a visual comparison between the situations may be made. A radar chart shows one axis for each aspect of a situation. Close to the center are the low values for the axis, and near the edge of the graph the high values are located. Such charts often show a current situation compared to some target. In the context of software systems, a certain solution or system can be considered as one situation and as such a radar chart offers a graphical display of the differences between actual and ideal performance and is useful for defining performance and identifying strengths and weaknesses of different solutions. Performance can depend on several aspects each represented in the radar chart. .We are going to visualize the different benchmark situations arised in the Case Study: MarsWold [1].

fig: results of the evaluation of the marsworld scenario for three different settings on three areas with different size.every setting was evaluated 50 times. the values in the parentheses express the speed up of time with respect to setting three for the respective area size.

Area Size	**Mean Time in Setting 1 (time units)**	**Mean Time in Setting 2 (time units)**	**Mean Time in Setting 3 (time units)**
Small	57.6 (+19%)	59.4 (+16%)	70.8
Medium	235.7 (+48%)	231.7(+49%)	449.4
Large	543.9 (+49%)	456.6 (+58%)	1071.6

In a This experiment, the prototype has been used to benchmark a distributed scenario called *MarsWorld*. In this scenario three different autonomous component types cooperate in order to explore ore on the Mars. The first type, called *sentry*, is responsible for analysing whether certain places contain ore resources. A second type, called *producer*, is in charge of exploiting detected ore resources. Finally, a type called *carrier* is responsible for transporting ore to a factory. This MarsWorld scenario has been benchmarked in order to evaluate the outcome of two different coordination strategies. In the first setting all autonomous component types are moving on the area and initially determining their next destination by random. Additionally, the autonomous components are also sharing all visited destinations. Therefore, they determine their following destination again randomly but exclude destinations that have already been visited by them or the other components. The second setting extended this coordination functionality with an additional behavior. As soon as a sentry has found ore resources on a position in the area it informsalso a carry agent, i.e. in addition to the producer agent which is also called as it was in the first setting. The idea of this behavior is to speed up the whole process. These two settings where compared to a third setting called basic setting. This setting did not contain the features of sharing already visited destinations neither the additional coordination functionality of setting two. Therefore, this third setting is well suited to benchmark the effect of the different coordination strategies mentioned previously.

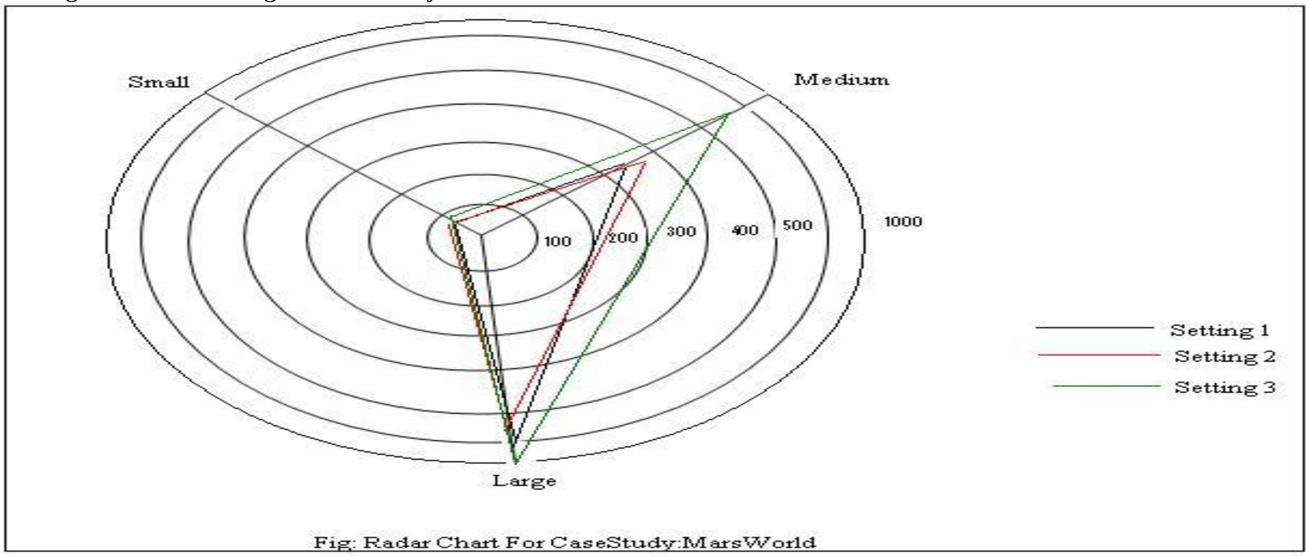

Fig: Radar Chart For CaseStudy:MarsWorld

Table I depicts the results from 50 evaluation runs for each setting. It shows the time, i.e. mean value, that was required to find, exploit and transport all ore deposits from the area. The table also reveals that the scenario has been evaluated for three areas with different size: small, medium and large. In general the results reveal the speed up of time that can be reached with the coordination strategies. For each evaluation of setting one and two the speed up is denoted in parentheses in table I with respect to the time required for setting three for the same area size. Even for a small area the speed up is significant and for an area with medium respective large size it is in fact very high. The results also reveal that the additional coordination function of setting two pays off only for large areas. Setting two can therefore be used as a benchmark for this application and other settings have to ”compete” against this configuration

All the information depicted in the table 1 was shown comprehensively in a single radar chart .By single glance we can say that a setting 3 can be observed as a default benchmark

.

IV. RADAR CHARTS IN THE ARCHITECTURE OF THE SOFTWARE FRAMEWORK FOR EVALUATING AND BENCHMARKING SELF-ADAPTIVE DISTRIBUTED SYSTEMS.

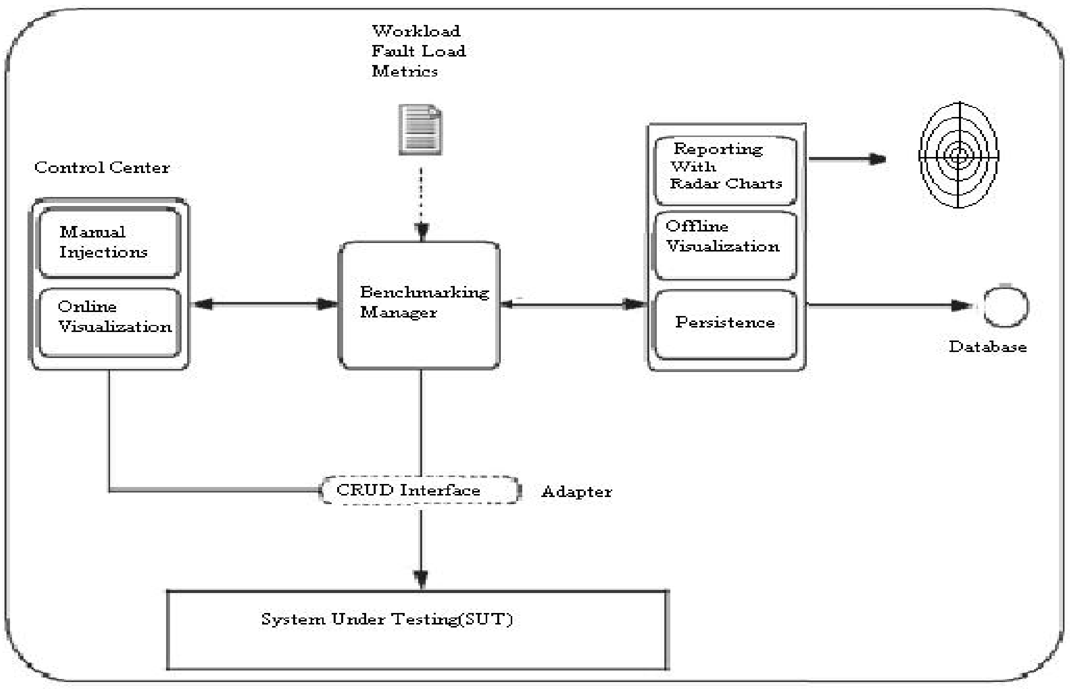

Fig: Usage of Radar Charts in The Architecture of the SoftwareFramework for Evaluating and Benchmarking SelfAdaptive Systems

It has been given Detailed description about the architecture in [1]. By taking the quick recall at the architecture,The *benchmarking manager* is responsible for all aspects that are related to the runtime management. It is responsible for the execution of the specified sequences at the right time. The benchmarking manager is also responsible for monitoring the termination condition of the conducted benchmark.For certain scenarios, it might be helpful to have a *control center* which gives (status) information about currently executed benchmarks. On the one hand, the control center offers an online visualisation component to obtain information about important metrics. benchmarks was reported here with the different metrics.,so for the easy visualization of the entire benchmarks Moreover, it can visualise the results of already conducted experiments.for easy view of the different benchmarks radar charts had been used here.

Also, the online visualisation offers the possibility to stimulate discussions between software developers and domain experts about the characteristics of the self-adaptive software system in early stages of the development process. On the other hand, the control center has a component called manual injection. It supports the manipulation of conducted benchmarks at runtime since it allows changing the configuration of the SuT via the CRUD interface.

Finally, the software architecture contains a *reporting component*. It offers the possibility to search for already conducted benchmarks and helps to identify settings that have already been benchmarked or that have still to be done.

Moreover, it can visualise the results of already conducted experiments. Obviously, the reporting component requires interaction with a database.They are very useful when a relatively small number of samples need to be compared and the number of variables or factors to look at is large.

We will be using radar charts in the reporting component to better support offline visualization.Since they are well suited to depict different metrics in the same diagram.

In this architecture the radar charts with the Reporting Component,this is mainly because, entire

Radar chart can be defined as arelevant way to judge functional adequacy of a self adaptive system. An ideal system is on the radar center because it outperforms the other in all the dimensions. Radar chart

Allow us to apply these metrics in overall evaluation and comparison approach. Because of these different solutions can be compared along with the multiple dimensions.

V. CONCLUSION

In this paper the main advantage is visualization of different metrics in the single radar chart, there by getting the benchmarks that are conducted for the self adaptive systems. As a Future works ,radar charts can be used in the online visualization also,along with the reusable metrics.

REFERENCES

1. Ante Vilenica & Winfried Lamersdorf ," Benchmarking and Evaluation Support for Self-Adaptive Distributed Systems" Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on Digital Object Identifier: 10.1109/CISIS.2012.115 Publication Year: 2012 , Page(s): 20 - 27 IEEE Conference Publications
2. B. Edmonds, "Using the experimental method to produce reliable self-organised systems," in Engineering Self-Organising Systems: Methodologies and Applications, ser. LNAI, S. e. a.Brueckner, Ed. Springer, 2004, no. 3464, pp. 84–99.
3. E. Kaddoum, M.-P. Gleizes, J.-P. George, and G. Picard,"Characterizing and evaluating problem solving self-* systems," in Future Comp., Service Comp., Cognitive, Adaptive,Content, Patterns, COMPUTATIONWORLD '09, 2009, pp.137 –145.
4. T. De Wolf and T. Holvoet, "Evaluation and comparison of decentralised autonomic computing systems," Dept. of Comp. Sc., K.U.Leuven, CW Rep. 437, Mar. 2006, accessedon 1-11-2011. [Online]. Available: https://lirias.kuleuven.be/ handle/123456789/131666
5. H. Madeira and P. Koopman, "Dependability benchmarking: making choices in an n-dimensional problem space," in Proceedings of the first Workshop on Evaluating and Architecting System Dependability, 2001.
6. P. Reinecke, K. Wolter, and A. van Moorsel, "Evaluating the adaptivity of computing systems," Perform. Eval., vol. 67, pp. 676–693, August 2010.
7. A. G. Ganek and T. A. Corbi, "The dawning of the autonomic computing era," *IBM Syst. J.*, vol. 42, pp. 5–18, 2003
8. M. Salehie and L. Tahvildari, "Self-adaptive software: Landscapeand research challenges," *ACM Trans. Auton. Adapt. Syst.*, vol. 4, pp. 14:1–14:42, May 2009.
9. R. Laddaga, "Self-adaptive software," December 1997dARPA Broad Agency Announcement, BAA-98-12.
10. J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.
11. A. Computing, "An architectural blueprint for autonomic computing." *White Paper*, vol. 36, no. June, p. 34 2006. [Online]. Available: http://users.encs.concordia.ca/~ac/ ac-resources/AC Blueprint White Paper 4th.pdf
12. T. DeWolf and T. Holvoet, "Emergence and self-organisation: a statement of similarities and differences," in *Proceedings ofthe International Workshop on Engineering Self-Organising Applications*. Springer, 2004, pp. 96–110.