Modeling Cloud Failure Data: A Case Study of the Virtual Computing Lab

Meiyappan Nagappan
Dept. of Computer Science
North Carolina State University
Raleigh, NC, USA
mnagapp@ncsu.edu

Aaron Peeler
Virtual Computing Lab
North Carolina State University
Raleigh, NC, USA
aaron_peeler@ncsu.edu

Mladen Vouk
Dept. of Computer Science
North Carolina State University
Raleigh, NC, USA
vouk@ncsu.edu

ABSTRACT

Virtual Computing Lab is a higher education cloud computing environment that on demand, allocates a chosen software stack on the required hardware and gives access to the customers, in this case NCSU students, faculty and staff. VCL has been in operation since 2004. An important component of the quality of the services provided by a cloud is the reliability and availability. For example, typical availability of the system exceeds 0.999, and reservation reliability is in the 0.99 range. VCL provides comprehensive information (provenance, logs, etc.) about its execution, its resources, and its performance. We mined the VCL log files to find out more about its reliability and availability, and the character of its faults and failures. This paper presents some of these results.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification—*Reliability,Statistical methods*; C.4 [**Performance of Systems**]: Reliability, Availability, and Serviceability

General Terms

Measurement, Reliability, Theory

Keywords

Cloud Computing, VCL Production Software Failures, Software Reliability and Availability

1. INTRODUCTION

General high-level architecture of a cloud environment is illustrated in Fig. 1 Every cloud needs to support a) some type of bare-metal or virtual resources (computational, storage, networking), b) collection of provenance and other information about its operation, failures, data, etc. (e.g., through logs), c) one or more user and management interfaces, d) help desk and maintenance infrastructure, e) appropriate security, privacy, license management, etc., f) authentication, authorization, and accounting, g) domain specific service (e.g., a Healthcare cloud would have a number of healthcare specific services), h) communications via network and one or more API-s for access across network, i) a data-bus (possibly multiple methods of access internal and external storage), and j) appropriate fault-tolerance and reliability characteristics.

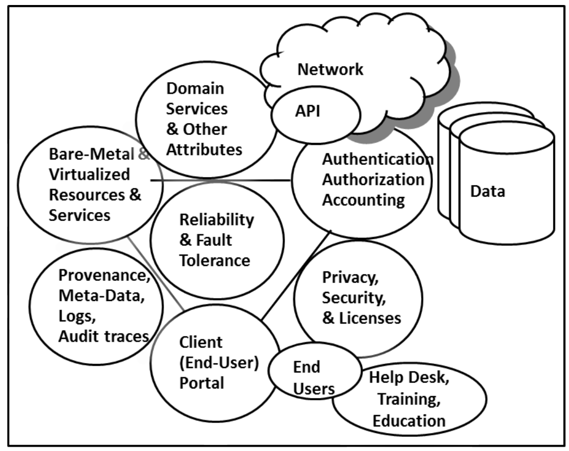

Figure 1: Cloud Computing Architecture

This paper is focused on a discussion of the reliability and availability [2], [9] of a production cloud operating at NC State University (NCSU) called Virtual Computing Laboratory (VCL) [1].

VCL [1] is an award-winning open source technology for delivery of cloud computing services developed in partnership with IBM and several other organizations. VCL has been in production operation since 2004. Its community is considerable and includes a number of NC and US higher-education institutions, as well as higher-educations institutions abroad. VCL is in pilot assessment for use in K-12 environments.

It is well known that network-based education requires highly reliable and fault-tolerance services. Plato and NovaNET [3] have have been offering reliability and failure assessment information to its users for over 50 years now [5], [7], [8], [14].

Armbrust et.al. [4], Buyya et.al. [6], and Vouk [16], have expressed the need for reliability in the cloud. They note that appropriate reliability and fault-tolerance is very important for the success of cloud computing.

In VCL, users request one or more software stacks called 'images'. VCL loads the requested image(s) on to the assigned hardware and after appropriate authentication and authorization gives access to the user. This is called a 'reservation'. At NCSU, VCL delivers about 250,000 dedicated sole-use 'image' reservations per year along with another 10.5 million High-Performance Computing (HPC) CPU hours per year.

In this paper we discuss VCL reliability and failure categories. Using VCL logs, we determined the times when failures occurred as well as their character. There are a number of failure and fault categories and we discuss some of them. Pro-active monitoring the cloud system provenance, performance and security is at the heart of all corrective and prevention actions. A good understanding of the roots of the problems and of the reliability profile of a cloud computing environment allows us to both better react to issues and mitigate them, as well as preventively manage them, and avoid them [12]. Some of the related actions include a) allocation of extra human maintenance resources at times when issues are predicted to arise due to capacity overload, security situations, and similar, b)"borrowing" of resources from other clouds, c) load balancing, etc.

1.1 Organization of this Paper

In Section 2 we provide more information about VCL. We also describe the format of the VCL logs and logging processes that are relevant to the current discussion. In Section 3 we discuss the nature of VCL failures and faults, and relevant descriptive and predictive modeling. In Section 4 we present the related work, and in Section 5 we conclude and point to future directions of research.

2. VIRTUAL COMPUTING LABORATORY

VCL "is an open source implementation of a secure production level on-demand utility computing and services oriented technology for wide-area access to solutions based on virtualized and bare-metal resources, including computational, storage and software resources" [1], [13], [17]. The infrastructure at NCSU has more than 2000 computers (most of them are IBM BladeCenter blades) and delivers diverse computing services to more than 40,000 users. There are close to 900 different software service stacks called 'images'. An image can range from a bare-machine hypervisor or operating system load, to virtual guest operating system loads, to any combination of the system and middlware and applications. Service can be individual and group based (e.g., for synchronized classroom or lab use), server-oriented, subclouds, and HPC. VCL also allows integration of external cloud services and hybrid cloud services. At NCSU VCL is operated as a private cloud, and all transactions and communications are logged. This includes networking, reservations and usage, security, resource provenance and performance logs, and other relevant information. Service reservations can be one-time or recurring, stateless or stateful, and they can be requested on-demand, or for some future time.

Fig. 2 illustrates VCL architecture. This architecture maps fully onto Fig. 1 general cloud components, i.e., VCL provides all required cloud elements. The users initially go

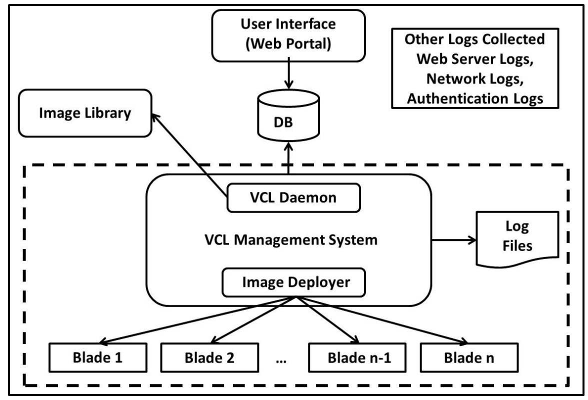

Figure 2: VCL Architecture

through a web portal to authenticate. Once authorized, VCL presents them with the choice of all the pre-built images available for them to reserve. The images available to choose from depend on the user privileges and category. The user then chooses one of the images, a cluster of images, or another mode of operation (e.g., stateful or stateless operation, creation of new service images using base-line images, etc.). They also request the time frame for which they want to use the resources. Their request is recorded in the database. Then a VCL checks on the availability of the resources and images, and hands the request over to one of its distributed resource management nodes (MN). Selected MN then loads the images (virtual or bare-metal) onto identified real or virtual hardware, and allows the user personalized access. Typically Linux based images are given direct access via ssh, or via the application service they are running, while Windows based desktops are are accessed using RDP protocol. Windows servers are accessed using appropriate application interface. Time-out, IP-locking, VLAN-ing and one-time and NCSU corporate passwords are used to secure the resources and services. VCL also has a protected private back-channel to all of its physical and virtual resources to load them, manage them and secure them. VCL can be used to seamlessly access resource on other VCL clouds, and commercial cloud services such as Amazon EC2, IBM Blue Cloud, and soon Azure.

2.1 VCL Management Node Log Format

VCL logs information at many levels. Centrally, its database has global information about all transactions (and failures) since the day VCL went into production (2004). However, each management node keeps extensive additional logs that allow optimization, debugging, forensics, management and recovery operations. Furthermore, each VCL image can be instrumented using open-source or proprietary resource monitoring agents.

In this paper we will focus on two sources - the relevant information from the global repository, and a sample of MN-based logs to illustrate the dynamics of the detailed processes and failures. Fig.3 illustrates the content of a record in the VCL daemon (or MN) log. It has a time stamp field, a unique identifier field, the caller information field, and information about the VCLd code that is emitting the message. The unique identifier is in turn comprised of three parts, namely the process ID, the (request ID,reservation ID) pair

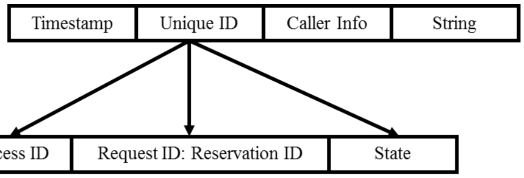

Figure 3: Partial Structure of VCLd (or MN) Log Record

and the state of the VCL system. We can determine which reservation a particular log line belongs to by extracting this detail.

MN log files are collected locally, in the same machine on which the VCLd daemon is executing, and so there is not much delay in collecting the information. Nevertheless, the time stamp is not the time at which the message is logged, but rather the time at which the logging method was called, i.e., when the particular event happened in the system. The MN clocks are synchronized to the NCSU campus time servers. The process ID is the thread ID that is currently handling this particular reservation. The (request ID, Reservation ID) pair is generated from the database at the time of image request and time of image reservation. They are two separate IDs since not all requests will result in a reservation. This can happen due to a failure after a request has been made but the reservation did not complete. The state of the system indicates which specific action of the reservation process the system is currently in. Using the combination of these three fields we can uniquely identify a particular action of a specific reservation. We need to identify each action in a reservation uniquely because each action can either complete execution successfully, or result in a failure. It is these failures that we are interested in. For each failure that may eventually become visible to the end-user (e.g., a failed reservation), there may be multiple action failures in the MN log.

The next field is the caller information field. This records which line of which file in the code makes the call to the logging method. This is used to identify which event has just executed. The last field in Fig.3, namely, the string field contains the actual log message. This is the field we inspect to determine if the action is a failure or not. The string contains the word 'CRITICAL' or 'WARNING' when a critical or warning type failure occurs.

Each of these fields are separated by the '|' (pipe) symbol. We read the log file line by line. Then we split each line into tokens based on the separator symbol. Once we have the tokens we examine the token for the string field to see if it has the case sensitive words 'CRITICAL' or 'WARNING'. If it does then we grab hold of the tokens for the timestamp and the unique identifier fields for that event. We write the time stamp to either the file with critical failures or the file with warning failures. Then we use the unique identifier and iterate through the log file till we reach the next action of the reservation. We do this because some times a specific action might have multiple warning or critical messages. This is due to the cascading effect of the failures.These warning and critical messages are often separated by mere seconds. Collecting them as separate failures would be incorrect as they are all of the same failing action of a reservation. Note that we consider failures that happen in different actions

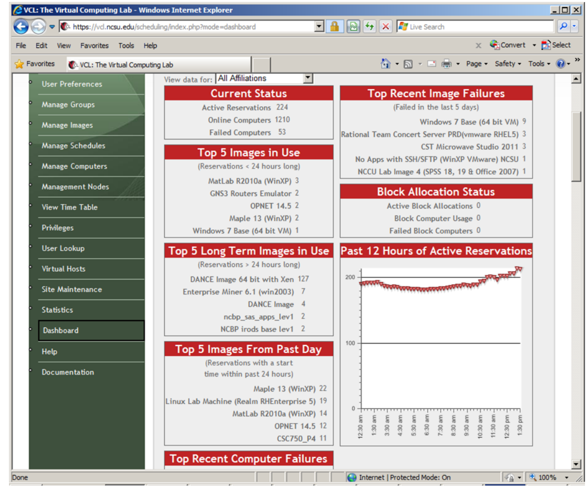

Figure 4: Dashboard for VCL Transactions

of the same reservation to be separate failures because it signifies that the reservation failed in multiple steps.

Note also that in the ideal implementation of this research, the analysis and prediction of failures would be done in real time. This paper however is an exploratory study, and here we extract and analyze failures from the log files after the fact.

2.2 VCL Database Logs

This set of logs provides global information about all VCL transactions at a particular installation. In the case of NCSU VCL, this encompasses 3 data centers, and users from NCSU and about 50 other NC institutions. While the internal MN logs are not open to public. A lot of useful information that can be gleaned from central transaction repository is presented to the VCL managers either through the Dashboard (Fig. 4) or to general public through the VCL Statistics interface tabs.

Dashboard provides, for example, information about the number of active reservations, on-line computers, and failed computers, top 5 images used, top recent computer failures, and top recent image failures. This allows quick and efficient insight into VCL operation, but it also facilitates mitigation and management activities.

It also points to a richness of faults and failure categories that exist in a cloud environment. Most of them are related to infrastructure issues, capacity issues, changes in operational profile (frequency of usage), and erroneous image to resource mappings. However, some are related to image building functions, and some (rare but present) to actual bugs in VCL codes.

Statistics display page allows a comprehensive insight into VCL operational profile over its life, into its reliability (and which images fail most often - usually capacity and image to hardware mapping related, but sometimes related to experiments with images themselves), and to failures that range from unavailability of resources (e.g., hardware unavailable, license unavailable, image off-line, etc.), to potential perception failures that have to do with speed of service (e.g., how long does it take to load an image), to actual failures on part of users to accept a reservation, and failures to prepare a computer for the end-user. Since VCL has built in fail-over

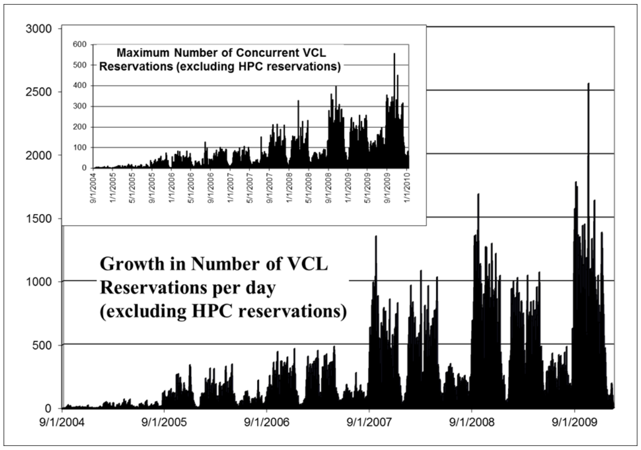

Figure 5: VCL Concurrent Reservations from September 2004 to February 2009

mechanisms, not all recorded failures end-up being seen by the end-user.

For example, one can learn that in between March 7, 2010 and March 7, 2011 there were close to 200,000 image reservation requests that consumed close to 490,000 computer-hours, that all but 7,247 of the reservations where for immediate use ("now") rather than being scheduled for later, and that end-user failed to receive requested resources (unavailable) in only about 0.55% of the cases, indicates an estimated reservation reliability of about 0.9945. In the same period of time there were no system-wide failures (i.e., at any time there always were some VCL resources available), however individual users may have experienced network or access related outages of a few minutes. A conservative estimate of the system availability is in excess of 0.999-s.

3. DISCUSSION OF USAGE AND FAILURES

3.1 Operational Profile

The key concept in any analysis of real-time dynamic large scale system, such as clouds, is its operational profile - the frequency of execution of its internal and external functions, services and operations, along with risks associated with the failure of any of those items [10].

VCL's operational profile is very complex. For example, its externally visible usage profile, on the general (non-HPC) side, is that of the number of reservations per day, and the number of concurrent reservations per day (refer Fig. 5). The latter is very important. If the concurrent capacity of the installation is below peak concurrent usage, users will experience failures in the form of unavailable resource messages. Today this concurrency threshold is as much as 700 to 800 real of virtual machines. Daily VCL usage is about 2 to 3 times that number.

When considering its primary service function (again on the non-HPC side), i.e. delivery of selected images, we see that, unsurprisingly, the "most popular" images are those that are used in large classes across campus, such as Maple, Matlab and MS Office. Table 1 illustrates some of the profile information for four most frequently used images in the last year. For example, long load times (usually due to the fact that the image has to be loaded from scratch onto a resource)

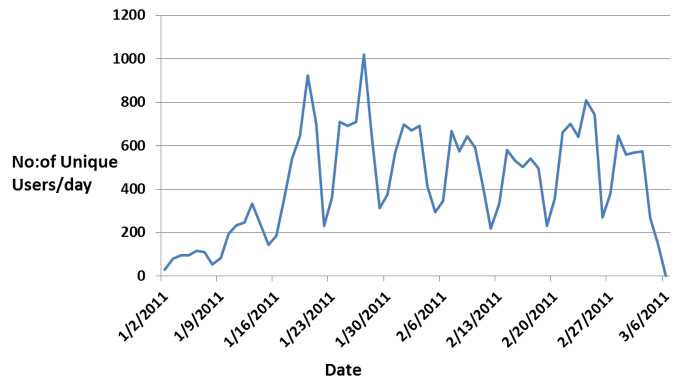

Figure 6: Number of Unique Users/Day

can be perceived as failures by users even if the reservations eventually succeed. On the average, our scheduling and pre-loading algorithms are about 80% effective in delivery very short reservation times. We are working on improving that through more intelligent pre-scheduling, virtualization and LUN-based booting.

To gain a more intimate insight into failures we take a peek "under the hood" of a management node. We do that by discussing data for 9 recent weeks from one of the MNs in one of our data centers. Fig. 6 illustrates the number of unique users of VCL (overall) on each day within our analysis time window. Fig. 7 illustrates the overall number of reservations requested over the same time period. We can see daily and weekly variation in usage. A peak during the middle of a week and tails during the weekends. Results are not unexpected and the graphs reflect well the start of the semester, holidays, etc. Fig. 8 on the other hand shows failure related messages recorded on one of the MN (only a fraction of the users were directed to that MN at any time, perhaps 10 to 20%). We see similar, but less regular peaks, multiple messages per failure, and some clustering consistent with increased overall usage and either hardware outages or a proportional increase in failures due to capacity limitations.

Table 2 summarizes some basic information about the MN log files we analyzed for the period 01/02/2011 to 03/06/2011. Each log file is between 600 and 1500 MB in size, and there are between 5.5 million 12 million events per log file. Each log line (or record) is an event that has been logged in the file by the VCL system when a pre-determined set of instructions were successfully executed. The string field in the log line contains the values for the variable parameters of that event. When an action doesn't complete in its entirety, then a failure has happened. This is indicated in the string field of the log line by the words 'CRITICAL' or 'WARNING' depending on which type of failure has occurred.

3.2 Modeling Failures

In general, if system usage is available (e.g., inservice time, number of users), and failure categories and times when they occur is available, then reliability and availability analysis of cloud-based environments should be fairly straightforward and comparable to that of other large scale systems (e.g., [7], [8], [9])

The specific failures of VCL that we investigates here are

Table 1: Top 4 image Reservation Statistics

Name	Reservations	Unique Users	Hours Used	<2 min load time	>=2 min load time	Failures
Maple 13 (WinXP)	34206	3668	45079	27061	7123	141(0.4%)
Linux Lab Machine	10375	1120	22085	10305	17	26(0.3%)
XINU (CSC501)	9093	148	19470	6832	2256	22 (0.2%)
E115 Office 2007	7281	959	11916	6627	653	21 (0.3%)

Table 2: Data on the log files of the case study

Name	Size(in MB)	No:of Events	Start Date	End Date	No:of Critical Failures	No:of Warning Failures
vcld.log.9	647.91	5,486,629	2011-01-02	2011-01-09	8	133
vcld.log.8	718.66	5,959,644	2011-01-09	2011-01-16	24	120
vcld.log.7	972.47	8,076,463	2011-01-16	2011-01-23	68	207
vcld.log.6	1,014.60	8,436,814	2011-01-23	2011-01-30	126	312
vcld.log.5	1,068.38	8,861,841	2011-01-30	2011-02-06	72	273
vcld.log.4	1,180.91	9,743,086	2011-02-06	2011-02-13	25	160
vcld.log.3	1,167.18	9,638,969	2011-02-13	2011-02-20	46	170
vcld.log.2	1,217.29	10,170,808	2011-02-20	2011-02-27	89	304
vcld.log.1	1,429.26	11,897,832	2011-02-27	2011-03-06	93	486

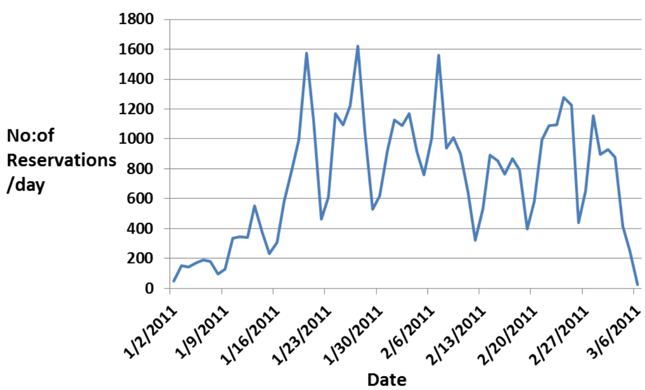

Figure 7: Number of Reservations/Day

those that occur when a user requests an image and the VCL system either fails to reserve the image, or takes more time than expected to make the reservation. When the user fails to get access to the requested image, it could be that the VCL system could not load the image on the chosen hardware, or it could not give the required access to a reserved image, etc.. These types of failures would be classified as 'Critical' failures. In other situations, the VCL system might make more than one attempt to load the image. This would be a 'Warning' type failure. Another example of this type of failure would be when the VCL system does not recognize a user, or information about a user, since it does not have the required data in its database. With reference to Fig. 8, we see more CRITICAL failures in the first two weeks (01/02/2011 to 01/16/2011) due to this. This can be also seen in the cumulative graph shown in Fig. 9.

Figs. 10 and 11 show similar graphs for 'Warning' failures. We see that the trends of 'Warning' failures closely match the trends of the 'Critical' failures.

These graphs indicate that fitting models to that data may be best driven by the past usage profiles. The more usage the more failures. Also start of a semester, or days just after a longer holiday, and days when large homeworks are due, will probably result in anomalies. The implication there is that once VCL is installed, the process failure rate is relatively constant and the total number of failures experienced during a day is typically commensurate with that installation's demand for certain type of images and concurrent usage of the resources. Special situations will exist. For example there can be instances when the number of reservations can be very high compared to the number of users. This is evident from the data on 02/07/2011, when the number of reservations requests is close to 1560 whereas the number of unique users are only 668. This means that some users were reserving more images than the usual. This could be due to a block (lab) reservation of many instances of an image. Another thing we observed is that the VCL software faults accounted for about 20% of the failures, while the remainder was for most part attributable to infrastructure related issues and resource exhaustions.

4. RELATED WORK

Buyya et.al. [6] discuss market oriented cloud computing. They believe that reliability in the cloud can be provided by a Service Level Agreement (SLA) oriented resource allocation. Each customer will require a different level of Quality of Service (QoS). Based on their needs, the cloud computing services provider can negotiate different SLAs. They note that cloud service providers must use market-oriented resource management to regulate the supply of demand of the cloud resources. They also must promote QoS based resource allocation mechanisms that uses the utility of consumers and providers for differentiating service requests.

This approach actually makes a lot of sense in the VCL context. Since most of the issues are resource driven, and in

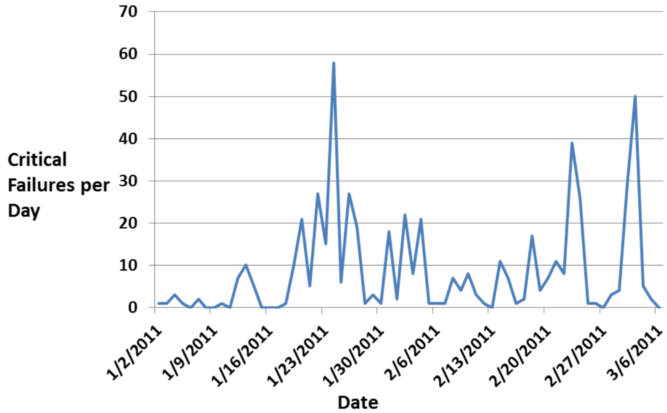

Figure 8: Critical Failures per Day for 9 weeks

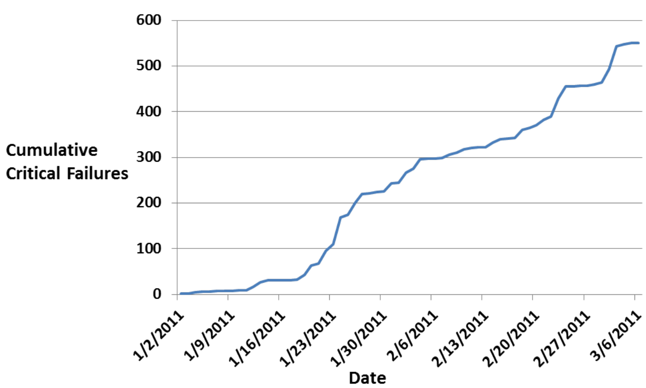

Figure 9: Cumulative Critical Failures for 9 weeks

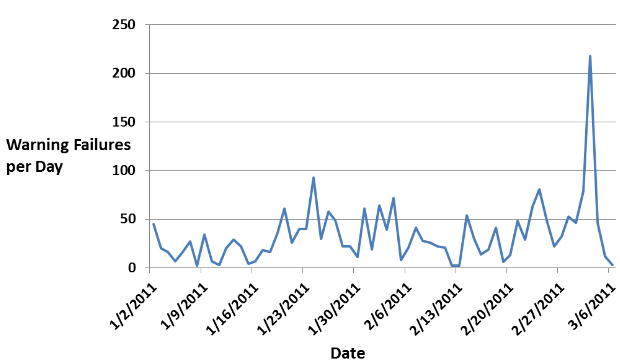

Figure 10: Warning Failures per Day for 9 weeks

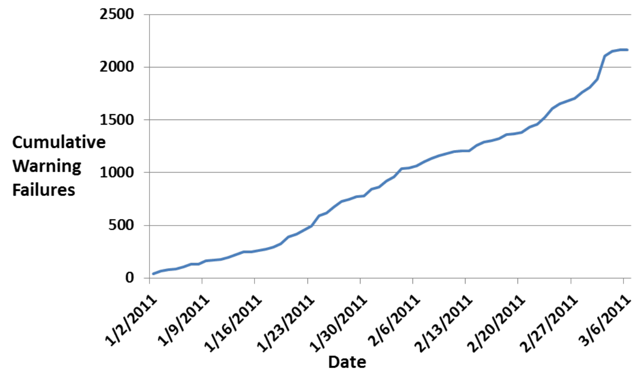

Figure 11: Cumulative Warning Failures for 9 weeks

reality VCL self-heals through very large redundancy of its resources, one can achieve a range of QoS and SLA goals.

Vouk [16] points out that cloud computing can improve reliability when compared to dedicated company specific self managed data centers. He points out that an important component of the reliability equation is handling of security faults and failures. Security faults and failures are a subset of the general set of system software and configuration faults, and VCL has an excellent record in that domain.

Armbrust et.al. [4] lists '10 obstacles and opportunities for cloud computing'. They believe that the rank 1 obstacle is reliability. They believe that cloud customers could face 'A single point of failure' if the particular cloud infrastructure is managed by a single company. They recommend that high availability customers use multiple cloud vendors to distribute the risk and provide fail over.

In the context of VCL, software reliability engineering literature [9] is probably less important than the hardware and resource exhaustion management literature [15], and software-based fault-tolerance literature (e.g., McAllister and Vouk [9] and references therein).

5. CONCLUSIONS AND FUTURE WORK

VCL is a flexible production cloud computing environment. Examination of its failures, faults and operational profile leads us to conclude that in this specific cloud computing environment, system software failures play a relatively small role. Most of the failures appear to be related to computational resource exhaustion, license exhaustion, and hardware failures. In a smaller number of instances mapping of images to appropriate hardware is the cause,i.e. accuracy of the mapping tables and site configuration management. Thus, in this context, classical software reliability engineering growth models do not appear to apply , or apply at 20% or so level. Instead, usage-profile-driven failure modeling (assuming relatively small and relatively constant process failure rate - in the range of 0.1% to 0.5%) may be an appropriate approach. This is more akin to the rare-event analysis discussed by Jones and Vouk in [9]. A bath-tub model may also apply in periods where major re-configuration of the system has occurred (in the case of VCL this is over longer holidays when we logically move our resources from general to HPC pool, and in the process reload all affected computers. While we have tried to use descriptive statistical models to describe cumulative failure rates in the context of adaptive logging [12], such approach has probably less predictive validity than simply multiplication of the usage profile curve by an appropriate constant failure rate and fraction of users on a particular MN.

In the plan is more exhaustive analysis of both global and management node logs, development of a failure and fault taxonomies, and development of hybrid reliability and availability models (a combination of hardware, fault-tolerance, and software models).

6. ACKNOWLEDGMENTS

We would like to thank Prasanth Anbalagan of North Carolina State University for his insights in to the regression analysis and discussion on reliability models, and to Josh Thompson for his help with data collection, and to both Josh Thompson and Andy Kurth for providing insights required for this study. This research was funded in part by the DOE grants DE-FC02-E25809 and DE-AC02-05CH11231, NSF grant NSF-0910767, ARO grant W911NF-08-1-0105, and AHRQ-NIH 5R21HS018218-02.

7. REFERENCES

[1] http://vcl.ncsu.edu/ (last accessed 01/20/2011)

[2] ANSI/IEEE, "Standard Glossary of Software Engineering Terminology," STD-729-1991, ANSI/IEEE, 1991.

[3] D. Alpert and D.L. Bitzer, "Advances in Computer-Based Education". Science, vol. 167, pp. 1,582-1,590, Mar. 1970.

[4] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. 2010. "A view of cloud computing". Commun. ACM 53, 4 (April 2010), 50-58.

[5] R.A. Avner, "Measuring Reliability of Networked Educational Computer Systems". CERL Evaluation Report, Univ. of Illinois, Urbana-Champaign, Aug. 1993.

[6] R. Buyya, C.S. Yeo, S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities," High Performance Computing and Communications, 2008. HPCC '08. 10th IEEE International Conference on , vol., no., pp.5-13, 25-27 Sept. 2008

[7] P. Dixit, "Quality of Service Modeling for Wide Area Network-Based Systems". PhD thesis, North Carolina State Univ., May 1998.

[8] P. Dixit, M.A. Vouk, D.L. Bitzer, and C. Alix, "Availability of NovaNET-A Wide-Area Network-Based Education System". Proc. ISSRE, IEEE CS Press, pp. 213-218, Oct. 1996.

[9] M.R. Lyu, "Handbook of Software Reliability Engineering," McGraw-Hill publishing, 1995, ISBN 0-07-039400-8.

[10] Musa J. D., 1975. "A Theory of Software Reliability and its Application", IEEE Transactions on Software Engineering, SE Ű 1(3), pp. 312-327.

[11] J.D. Musa, and K. Okumoto, 1984, "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement", Proceedings of the 7th International Conference on Software Engineering, Orlando, pp. 230-238.

[12] M. Nagappan, M.A. Vouk, "Adaptive Logging: A Case Study of Logs from a Cloud Computing Environment". In the Fast Abstracts track of International Symposium on Software Reliability Engineering, 1-4 Nov, 2010, San Jose, California.

[13] H.E. Schaffer, S.F. Averitt, M.I. Hoit, A. Peeler, E.D. Sills, M.A. Vouk, "NCSU's Virtual Computing Lab: A Cloud Computing Solution," Computer, vol.42, no.7, pp.94-97, July 2009

[14] E.R. Steinberg, "Computer-Assisted Instruction-Synthesis of Theory". Practice and Technology, Lawrence Erlbaum Assoc., Hollsdale, N.J., 1991.

[15] K.V. Vishwanath and N. Nagappan, "Characterizing Cloud Computing Hardware Reliability". In Proceedings of the 1st ACM symposium on Cloud computing (SoCC '10), pp. 193-204.

[16] M.A. Vouk, "Cloud computing - Issues, research and implementations," 30th International Conference on Information Technology Interfaces, vol., no., pp.31-40, 23-26 June 2008.

[17] M. Vouk, A. Rindos, S. Averitt, J. Bass, M. Bugaev, A. Peeler, H. Schaffer, E. Sills, S. Stein, J. Thompson, M.P. Valenzisi, "Using VCL Technology to Implement Distributed Reconfigurable Data Centers and Computational Services for Educational Institutions". IBM Journal of Research and Development, Vol. 53, No. 4, pp. 2:1-18, 2009