Cloud Solutions for Bug Reporting

Pano Gushev[1], Ana Guseva[2], Sasko Ristov[3] and Marjan Gusev[4]

Abstract—**Bug reporting software deals with software bugs and helps the overall software development process. Most of the existing solutions are free and self-hosted oriented. A prospective user should implement a new version on its own server, whenever a software project is about to start. A cloud oriented solution solves the problems that are initiated with enabling an efficient environment, by setting only one implementation and on demand usage with centralized access. Each upgrade or a new version is then automatically distributed to all users. In this paper we overview recent trends in cloud solutions for bug reporting.**

***Index Terms—Bug reporting*; *Ticket tracking*; *Issue tracking*; *Software Engineering*; *Cloud computing*; *Software as a Service*.**

I. Introduction

> Program testing can be used to show the presence of bugs, but never to show their absence!
> *Edsger Dijkstra* [1]

Writing software has evolved into a sophisticated software engineering profession. Most demanding trends address the efforts to maximize the software quality and the creation process. Bugs are those software malfunctions, which define an aspect of a quality, defined via various parameters, like measurable quantity of categorized bugs, customer satisfaction, etc.

Bug reporting software is a basic tool for communication between the supplier and customer. A simple analysis shows that each software development company uses a kind of a bug reporting software, to improve communication to the customers, having a direct impact on the quality of the software. It can also be used inside the company for the realization of internal processes.

Some of the web-based bug reporting systems were launched in the late 90's. Today, there are a lot of open source bug reporting systems, which companies can download and install on their servers (Self-Hosted). These systems can be developed in different frameworks, so the companies must have the appropriate hardware and software architecture to use them. A lot of them have a rather complex installation process, so the company needs to employ an educated person with sufficient knowledge of the framework used or sufficient experience in installing such a system. It also includes new costs, like maintenance of the system, which requires human involvement. The idea of cloud enabled bug reporting solution is not new, but there are still not enough fully developed cloud solutions. In this paper we give an overview of the basic concepts for realization of a cloud bug reporting solution, and present relevant architecture and design issues.

Companies can benefit if they migrate their bug reporting systems onto the cloud since it offers a good resource flexibility and scalability, storage, computational requirements and network access, and most importantly, lower cost. The Cloud lowers the disk space requirements and enables the latest software versions and monitoring of the installation progress in each of the cloud service models.

A careful consideration should be taken when migrating onto a cloud, especially about deployment model and service layers. The cloud architecture offers a solution for each cloud service layer from IaaS (Infrastructure as a Service), PaaS (Platform as a Service) to SaaS (Software as a Service). In this paper we analyze relevant issues for cloud hosting and use infinite cloud resources dynamically to reduce the costs and to provide better performance.

The rest of the paper is organized as follows. Section II presents related work to the cloud computing issues and bug reporting essential concepts. The challenges of a modern bug reporting system and its architecture are presented in Section III, along with the bug reporting cloud solution software engineering. We discuss the benefits of the cloud bug reporting solution and compare several existing architectures and organizations in Section IV. The final Section V is devoted to conclusion and future work.

II. Related work

Bug reporting plays an important role in the testing environment of a software project.

Bertolino and Marchetti [3] observe that testing is a crucial part of the software lifecycle, and recent trends in software engineering provide evidence of the importance of this activity during the development process. Testing activities have to start already at the requirements specification stage, with early planning of test strategies and procedures, and propagate down, with derivation and refinement of test cases, along these various development steps, starting from the code-level stage, at which the test cases are eventually executed, and even after deployment, with logging and analysis of operational usage data and customers reported failures.

A lot of bug reporting systems have been developed until now, like Bugzilla [6], Bug Tracker .NET [21], Eventum [26], FogBugz [7], Fossil [27], Jira [10], Mantis [23], OTRS [25], Pivotal Tracker [22], Rally Community Edition [5], Redmine [8], Request Tracker [24], Team Foundation Server [9], The Bug Genie [28], Trac [4], Unfuddle [11], WebIssues [29]. Most

[1] P. Gushev is with Innovation Dooel, Vostanichka 118-18, 1000 Skopje, Macedonia from 2008, Email: pano.gushev@innovation.com.mk

[2] A. Guseva is with Innovation Dooel, Vostanichka 118-18, 1000 Skopje, Macedonia from 2012, Email: ana.guseva@innovationl.com.mk

[3] S. Ristov is with University Ss Cyril and Methodius, Faculty of Computer Science and Engineering from 2010, Rugjer Boshkovik 16, PO Box 393, 1000 Skopje, Macedonia, Email: sashko.ristov@finki.ukim.mk

[4] M. Gusev is with University Ss Cyril and Methodius, Faculty of Computer Science and Engineering from 1989, Rugjer Boshkovik 16, PO Box 393, 1000 Skopje, Macedonia, Email: marjan.gushev@finki.ukim.mk

of them integrate an issue tracking software or built add-on features, like wikis, on-line collaboration, etc.

An extensive overview and comparison of bug reporting systems or issue tracking systems may be found on Internet, or in relevant literature [12], [13] etc. Deeper analysis and bug classification is reported by Zaman et al. [14], or Lal and Sureka [13], etc. However, there is no study that covers cloud aspects of the bug reporting solutions.

III. Cloud Bug Reporting Architecture

Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services [15].

Fuhrt and Escalante [16] consider that cloud computing can be defined as a new style of computing in which dynamically scalable and often virtualized resources are provided as services over the Internet. Cloud computing has become a significant technology trend, and many experts expect that cloud computing will reshape information technology (IT) processes and the IT marketplace. With the cloud computing technology, users use a variety of devices, including PCs, laptops, smartphones, and PDAs to access programs, storage, and application-development platforms over the Internet, via services offered by cloud computing providers. Advantages of the cloud computing technology include cost savings, high availability, and easy scalability.

The underlying technologies of cloud computing have been in use in some kind of form for decades [17]. For example, virtualization, arguably the biggest technology driver behind cloud computing is almost 40 years old. Virtualization has a long history, starting in the mainframe environment and arising from the need to provide isolation between users [18]. Modern computers are sufficiently powerful to use virtualization to present the illusion of many smaller virtual machines (VMs), each running a separate operating system instance [19].

Figure 1, adapted from Voas and Zhang [20], shows six phases of computing paradigms, from dummy terminals/mainframes, to PCs, networking computing, to grid and cloud computing. In phase 1, many users shared powerful mainframes using dummy terminals. In phase 2, stand-alone PCs became powerful enough to meet the majority of users' needs. In phase 3, PCs, laptops, and servers were connected together through local networks to share resources and increase performance. In phase 4, local networks were connected to other local networks forming a global network such as the Internet to utilize remote applications and resources. In phase 5, grid computing provided shared computing power and storage through a distributed computing system. In phase 6, cloud computing further provides shared resources on the Internet in a scalable and simple way. Comparing these six computing paradigms, it looks like cloud computing is a return to the original mainframe computing paradigm. However, these two paradigms have several important differences. Mainframe computing offers finite computing power, while cloud computing provides almost infinite power and capacity. In addition, in mainframe computing dummy terminals acted as user interface devices, while in cloud computing powerful PCs can provide local computing power and cashing support.

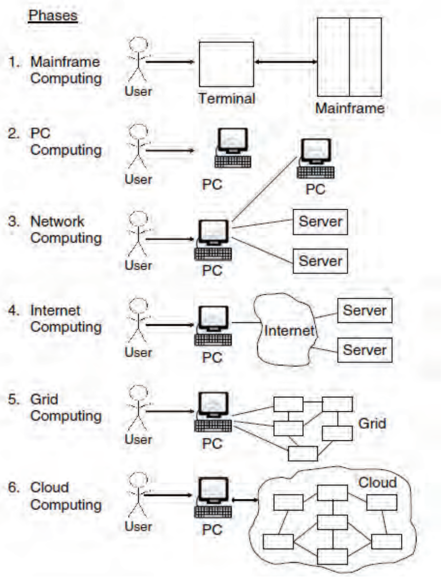

Fig. 1: Six Computing Paradigms

Companies can choose between two types of bug reporting systems: Self-Hosting and Cloud computing solutions.

Most of the bug reporting software solutions are build for self-hosting, meaning that the software supplier is required to install it on its own server. If self-hosting is realized on a rented server, then we address a solution with hosted infrastructure (IaaS) or platform (PaaS) as a kind of cloud solution, if the solution offers a possibility to share it among different customers or companies. On the other hand, sometimes the customer would like its own bug reporting software, especially in the case when this company starts several software projects with one or various suppliers.

Figure 2 shows a realization of a self-hosted bug reporting system. Each company that wants to use the system must have a full copy of the bug reporting system installed on their own hardware and software architecture. It can be an open source solution or own developed product. The access to the system is within the company perimeter, or it may be realized as standard web-access.

Sometimes it is possible to integrate this solution with other systems, i.e. Active Directory for user authentication. Furthermore, it offers possibilities to implement non-web-based client applications. But this system also comes with higher installation and technical support costs. When a company has developed their own bug reporting system, it probably required a lot of resources and increased the costs. This type of system requires occasional maintenance, especially in backing data source.

Figure 3 shows a cloud architecture of bug reporting system. In this case, the bug reporting system is hosted on cloud. Such an approach allows the system to be accessed from any company and from any place.

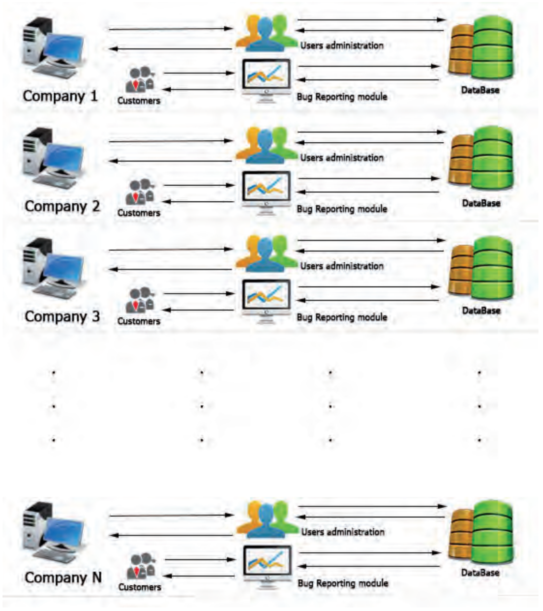

Fig. 2: Self-hosted bug reporting system

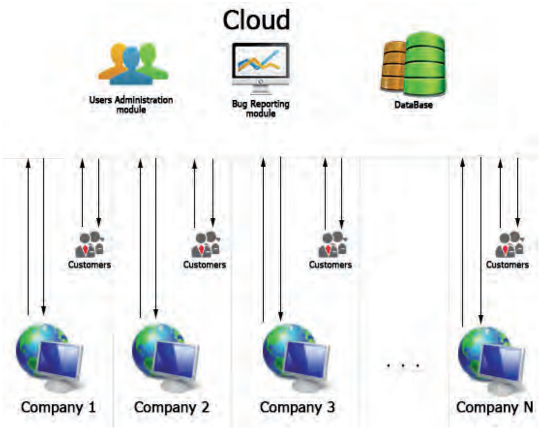

Fig. 3: Cloud bug reporting system architecture

Model	Choose	Install	Opportunity
Self-Hosted	buy own server resources and access capacity	own platform and application	client-server model
IaaS	server resources and access capacity	own platform and application	client-server model
PaaS	platform, server resources, access capacity	own application	application framework
SaaS	access capacity	-	business functionality

TABLE I: Features of various service models

No special hardware and software architecture are needed for companies to use this system, they only need Internet access and a web browser. This means that the users will not have capital investments, and they would rather pay only the running operating costs. The cloud architecture releases the companies from worries about the routine maintenance, backups and upgrades. All the security of the system is solely the responsibility of the vendor. This approach is cheaper from self-hosting systems because it does not require up-front investment. No capital expenditure is required. Users pay for services and capacity of resources used as they need them. In this type of bug reporting systems, companies do not have access to the code, so they can't customize the system as they wish, but this bug reporting system offers a lot of customization.

Bug reporting cloud solution is platform independent in case of SaaS and IaaS service models or installed on a specific platform in case of PaaS service model. On the other side, this solution can be offered as SaaS service, so the cloud provider will take care of the infrastructure agent and provide relevant number of resources.

The bug reporting solution can be hosted on appropriate IaaS or PaaS cloud service layers. The communication with cloud controllers is scalable and elastic. This solution allows a company to have a bug reporting system with low costs. In traditional self-hosting environment, companies must use a lot of their resources and spend a lot of time to build hardware and software architecture. Even more, not only will the equipment be underutilized most of the time, but it will spend a lot of electricity power unnecessary. On the other hand, with the cloud solution they only need an access to the internet and a web browser.

Table I presents features of different cloud service models.

IV. Discussion

Bug reporting system as a cloud computing solution offers a lot of advantages for companies. This type of solution offer the companies to save money, resources, time for developing etc. The final goal is to build all the functionalities which any other bug reporting system can, but at much lower cost. Companies can forget about expensive hardware and software architecture, installations, upgrades etc. They only need a web browser, which means anyone can use this system. They can have all upgrades withhold compatibility issues. Bug reporting system offers a lot of customization to fulfill the requirements of different types of companies. Companies can be sure that someone else takes care of the security of the system and its maintenance. Being scalable and elastic is also one very important issue that one should take care when choosing a proper cloud solution.

We have made an analysis of implementation issues for several bug reporting systems covering the cloud implementation aspects. Table II presents how a typical implementation can be hosted on cloud.

A typical cloud solution can be hosted on all three cloud service layers, i.e IaaS, PaaS and SaaS. The solution should be capable to instantiate as many VM instances as enabled by hardware resources in the computer center.

A very important aspect is usage of interoperability, by setting standards for interoperable systems. This means that the cloud solution should have a possibility to save all test

Product	Hosting Model	Purchase Cost	complexity
Trac [4]	Self-Hosting	Free	Low
Rally Community Edition [5]	Cloud	Free up to 10 users	High
Bugzilla [6]	Self-Hosting	Free	Low
Bug Tracker .NET [21]	Self-Hosting	Free	Low
FogBugz [7]	Self-Hosting and Cloud	$25/user per month SaaS; $190/user for self-hosted solution	Med
Redmine [8]	Self-Hosting	Free	Med
Team Foundation Server [9]	Self-Hosting	10 users: about $300/user	High
Jira [10]	Self-Hosting and Cloud	$10 for self-hosted solution for up to 10 users; SaaS starts at $15/user/-month	High
Unfuddle [11]	Cloud	10 users: $9/mo/user	Low
Pivotal Tracker [22]	Cloud	Free	Med
Mantis [23]	Self-Hosting	Free	Med
Request Tracker [24]	Self-Hosting	Free, pay for support	High
OTRS [25]	Self-Hosting	Free	Low
Eventum [26]	Self-Hosting	Free	Med
Fossil [27]	Self-Hosting	Free	Med
The Bug Genie [28]	Self-Hosting and Cloud	5 users $15 etc. free for self-hosting	Low
WebIssues [29]	Self-Hosting	Free	Low

TABLE II: Comparison of different cloud enabled solutions for bug reporting

cases, or other bug reporting and ticket tracking activities and transfer it to a new system. This might sound very futuristic, but it should be soon realized, in a form of API, using XML files. Transfer of users, as discussed earlier is another form of interoperability.

V. Conclusion

There are several challenges for cloud bug reporting system that initiate motivation to replace the traditional self-hosted bug reporting systems. For example, one of the most important challenges is to offer users administration module and bug reporting module for all the companies in a centralized system. Then the activities concerning the ticket tracking and test case processing can be scheduled to different virtual machines and offer a high scalability and elasticity of the cloud solution. In near future we plan to realize a new cloud solution capable to handle different workloads, by specifying details of its architecture and organization that improves the overall performance and reduces the costs.

In this paper we have analyzed the most used bug tracking solutions analyzing the possibility to use them as cloud solutions. the systems are identified by a possible their deployment layer model, as self-hosting solutions or cloud solution. A brief discussion is given of implementation issues concerning the IaaS, PaaS and SaaS layers.

Besides the benefits, the cloud solution has several disadvantages, like the complicated user management system, ability to exchange user sensitive data, designing a system with proper dimension which is efficiently used, ability to exchange bug relevant data, etc. They mainly concern interoperability issues, flexibility to configure own implementation, and elasticity of the solution. In the future we plan to develop a SaaS solution that deals with these issues and open questions.

References

[1] E. Dijkstra, "7.4 structured programming," *Software Engineering Techniques*, p. 65, 1970.

[2] I. Sommerville, *Software Engineering. (9th edition).* Addison Wesley, May, 2010.

[3] A. Bertolino and E. Marchetti, "A brief essay on software testing," *Software Engineering, The Development Process. Wiley-IEEE Computer Society Press,*, 2005.

[4] Trac. (2013). [Online]. Available: http://trac.edgewall.org/

[5] Rally Developer. (2013) Rally Community Edition. [Online]. Available: http://www.rallydev.com/product-features/rally-community-edition

[6] Bugzilla. (2013). [Online]. Available: http://www.bugtracker.com/

[7] Fog Creek. (2013) FogBugz. [Online]. Available: http://www.fogcreek.com/fogbugz/

[8] Redmine. (2013). [Online]. Available: http://www.redmine.org/

[9] Microsoft. (2013) Team Foundation Server. [Online]. Available: http://msdn.microsoft.com/en-us/vstudio/ff637362.aspx

[10] Atlassian. (2013) Jira. [Online]. Available: http://www.atlassian.com/software/jira/overview

[11] Unfuddle. (2013). [Online]. Available: https://unfuddle.com/

[12] I. V. Vlasceanu and C. Bac, "A study concerning the bug tracking applications," INF Department, TELECOM and Management, SudParis, Tech. Rep., 2008.

[13] S. Lal and A. Sureka, "Comparison of Seven Bug Report Types: A Case-study of Google Chrome Browser Project," in *2012 IEEE 19th Asia-Pacific Software Engineering Conference*, 2012.

[14] S. Zaman, B. Adams, and A. E. Hassan, "A qualitative study on performance bugs," in *Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on*. IEEE, 2012, pp. 199–208.

[15] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[16] B. Furht and A. Escalante, *Handbook of cloud computing*. Springer, 2010.

[17] B. Williams and G. Reese, *The Economics of Cloud Computing*. Cisco Systems, 2012.

[18] S. Crosby and D. Brown, "The Virtualization Reality: Are hypervisors the new foundation for system software?" *ACM queue – Computer Architecture*, vol. 4, no. 10, Dec. 2006. [Online]. Available: http://queue.acm.org/detail.cfm?id=1189289

[19] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.

[20] J. Voas and J. Zhang, "Cloud computing: new wine or just a new bottle?" *IT professional*, vol. 11, no. 2, pp. 15–17, 2009.

[21] Bug Tracker Software. (2013). [Online]. Available: http://www.bugzilla.org/

[22] Pivotal Tracker. (2013) Simple, agile project management software. [Online]. Available: http://www.pivotaltracker.com/

[23] Mantis Bug Tracker. (2013). [Online]. Available: http://www.mantisbt.org/

[24] Best Practical. (2013) Request Tracker. [Online]. Available: http://bestpractical.com/rt/

[25] Open Technology Real Services. (2013) Open Source Ticket Request System. [Online]. Available: http://bugs.otrs.org/index.cgi

[26] Eventum. (2013) Issue and bug tracking system. [Online]. Available: http://eventum.com/

[27] Fossil. (2013) Bug Tracking and Wiki. [Online]. Available: http://fossil-scm.org/index.html/doc/trunk/www/bugtheory.wiki

[28] The Bug Genie. (2013) Friendly issue tracking. [Online]. Available: http://www.thebuggenie.com/

[29] WebIssues. (2013) Issue tracking and team collaboration system. [Online]. Available: http://webissues.mimec.org/