Enterprise file synchronization and sharing services for educational environments in case of disaster

Servicios de sincronización y almacenamiento de archivos para entornos educativos en caso de desastre

Serviços de sincronização e armazenamento de arquivos para ambientes educativos em caso de desastre

Ana Isabel Delgado-Domínguez*
Walter Marcelo Fuertes-Díaz**
Sandra Patricia Sánchez-Gordon***

Fecha de recepción: 1 de septiembre de 2017
Fecha de aprobación: 2 de diciembre de 2017

Abstract

Cloud computing is an emerging solution that responds to the concept of Smart University; it aims at offering an intelligent environment of business continuity for the actors of an educational center. This research offers a recovery plan of educational services in case of disaster, through an action research, which analyzed free software for cloud computing, focusing on Enterprise File Synchronization and Sharing (EFSS). To achieve this, the implementation was placed in a local scenario (Linux Apache, MySQL, PHP, LAMP), and stress tests were performed on three applications: Nextcloud, Seafile and Pydio. Nextcloud had more consistent and better results than the other two applications; however, it lacks a system that allows synchronizing two Nextcloud instances. To solve this, we developed a routine aimed at providing an environment that monitors the hot site where the application is hosted and, from time to time, synchronize the instance to avoid data loss during disaster events. Afterwards, we configured a second application on a cold site that is alert to a possible service breakdown, so it can respond and sent immediate alerts. Finally, the usability of the routine was evaluated, and the disaster recovery plan for the EFSS was assembled, to offer a continuity of the educational services that are running in these environments.

Keywords: business continuity; cloud computing; disaster recovery; educational technology; information technology; open source.

Resumen

Las nubes computacionales son una solución emergente que responden al concepto de Smart University, para proporcionar un entorno inteligente de continuidad del negocio para los actores de un centro educativo. Esta investigación propone un plan de recuperación de servicios educativos en caso de desastres, aplicando la metodología de Investigación-Acción, que incluye un análisis de nubes computacionales de software libre, al

* M. Sc. Escuela Politécnica Nacional (Quito, Ecuador). ORCID: 0000-0003-2348-2260. ana.delgado@epn.edu.ec.
** Ph.D. Universidad de las Fuerzas Armadas ESPE (Sangolquí, Ecuador). ORCID: 0000-0001-9427-5766. wmfuertes@espe.edu.ec.
*** Ph.D. Escuela Politécnica Nacional (Quito, Ecuador). ORCID: 0000-0002-2940-7010. sandra.sanchez@epn.edu.ec.

enfocarse en los sistemas de Sincronización y Uso Compartido de Archivos Empresariales (EFSS). Para llevarlo a cabo se diseñó e implementó un escenario local (Linux Apache, MySQL, y PHP, LAMP). Para su evaluación y validación se realizaron varias pruebas de estrés en tres aplicativos: Nextcloud, Seafile y Pydio. Entre los hallazgos se evidenció que Nextcloud tuvo resultados consistentes por encima de las dos opciones restantes; sin embargo, esta solución no tiene un sistema que le permita sincronizar dos instancias de Nextcloud. Para solucionarlo, se desarrolló una rutina con el objetivo de proporcionar un ambiente que monitoree un *hot site* donde está alojado el aplicativo en producción y, cada cierto tiempo, realice sincronizaciones de la instancia para evitar la pérdida de información en caso de desastres. Luego, se configuró un segundo aplicativo en un *cold site* que está atento ante una posible caída del servicio, para su respuesta y alerta inmediata. Por último, se evalúo la usabilidad de la rutina y se ensambló un plan de recuperación de desastres para las EFSS, a fin de ofrecer una continuidad de los servicios educativos que se gestan en estos entornos.

Palabras clave: continuidad del negocio; nube computacional; software de código abierto; recuperación de información; tecnología educacional; tecnología de la información.

Resumo

As nuvens computacionais são uma solução emergente que respondem ao conceito de Smart University, para proporcionar um ambiente inteligente de continuidade do negócio para os atores de um centro educativo. Esta pesquisa propõe um plano de recuperação de serviços educativos em caso de desastres, aplicando a metodologia de Pesquisa-Ação, que inclui uma análise de nuvens computacionais de software livre, ao enfocar-se nos sistemas de Sincronização e Uso Compartilhado de Arquivos Empresariais (EFSS). Para levá-lo a cabo desenhou-se e implementou-se um cenário local (Linux Apache, MySQL, e PHP, LAMP). Para sua avaliação e validação realizaram-se várias provas de estresse em três aplicativos: Nextcloud, Seafile e Pydio. Entre os achados evidenciou-se que Nextcloud teve resultados consistentes acima das duas opções restantes; porém, esta solução não tem um sistema que lhe permita sincronizar duas instâncias de Nextcloud. Para solucioná-lo, desenvolveu-se uma rotina com o objetivo de proporcionar um ambiente que monitore um hot site onde está alojado o aplicativo em produção e, cada certo tempo, realize sincronizações da instância para evitar a perda de informação em caso de desastres. Logo, configurou-se um segundo aplicativo em um cold site que está atento ante uma possível queda do serviço, para sua resposta e alerta imediata. Por último, avaliou-se a usabilidade da rotina e elaborou-se um plano de recuperação de desastres para as EFSS, a fim de oferecer uma continuidade dos serviços educativos que se gestam nestes ambientes.

Palavras chave: continuidade do negócio; nuvem computacional; software de código aberto; recuperação de informação; tecnologia educacional; tecnologia da informação.

Cómo citar este artículo:
A. I Delgado-Domínguez, W. M. Fuertes-Díaz, and S. P. Sánchez-Gordon, "Enterprise file synchronization and sharing services for educational environments in case of disaster," *Rev. Fac. Ing.*, vol. 27 (47), pp. 81-91, Jan. 2018.

I. Introduction

The inclusion of Information and Communication Technologies (ICT) in education has taken a slow and uneven pace, both for their economic and human resources. As a consequence, in disaster situations there is no response that allows continuing the processes linked to administration, documentation, tracking, reporting, and delivery of educational courses. Given this, the computational cloud promises to reduce costs and offer high availability and long-term continuity [1]. The cloud is considered a model of flexible delivery of ICT services that provide systems and networks with high transfer rates [2].

The computational cloud arises from the need to build less complex IT infrastructures in comparison with the traditional technological schemes [3-6], in which the technicians install, configure and improve the software systems, hence, the assets of infrastructure are inclined to quickly become obsolete. Therefore, using these computing platforms is a solution for IT users, as an intelligent technology that responds to the Smart Education model, by offering a robust infrastructure environment.

The vision of Smart University deploys a set of services that focus on large-scale interactions, conceiving the university as a deeply dynamic and innovative place. To achieve this, Smart Education has its foundations on smart devices and emerging technologies [7] that respond to mobile learning. When using devices, it focuses on learner mobility, in contrast to traditional types of education [8]. Ubiquitous technology focused on learning can be used anytime and anywhere, without limitations of time, location, desktop, or mobile environments. Thus, intelligent technologies such as the computational cloud promoted the appearance of Smart Education. In this way, the advent of computational cloud has generated additional options for educators and students, providing them with the means to express their research, studies, and creativity in a distinctive way [9]. Its application not only alleviates the burden of educational institutions to manage the complex IT infrastructure, but also leads to great cost savings [10].

The present study focused on the recovery of services in educational environments, such as storage, communications, sharing, and file synchronization, by combining elements of the computational cloud supported by the Smart Education theory, such as the Synchronization and Use systems: Shared Enterprise File Synchronization and Sharing (EFSS). This allows us to respond to the particularities observed in disaster situations, such as the devastating earthquake of April 16, 2016, which left the education sector in the province of Manabí (Ecuador) out of operation due to the lack of a recovery plan in these eventualities of force majeure.

The rest of the article has been organized as follows: section 2 describes the state of art; section 3 synthesizes the applied research methods and techniques, as well as the activities carried out to bring the work to a successful conclusion; section 4 details the quantitative evaluation of the EFSS platforms; section 5 explains the start-up of the experiment; section 6 describes the experimental results of the EFSS implementations, and the execution of the developed routine in two EFSS instances, the measurement of the quality of use of the routine, and the discussion of the results obtained; finally, section 7 presents the conclusions and lines of future work based on the obtained results.

II. Related work

In various sectors, the use of technology based on file sharing via Internet, among the users of an organization and in collaboration with others, is becoming more and more prevalent [11]; therefore, it requires a careful selection of the solution in terms of administration, security, and costs.

In education, cloud-computing services provide a faster recovery and discovery of information, allowing students to store and share documents in a more flexible environment, and remote access to materials between students and instructors [12]. Computational clouds, in services such as Google Drive, Dropbox, Sky-Drive, and iCloud, offer the user the possibility of storing, reviewing, and accessing files synchronized among various devices [12], with a limiting use license that requires, among other aspects, a subscription fee and content restrictions. The main educational activities conducted in the computational cloud focus on discussing, planning, and using the interactive applications and services that are carried out in colleges and universities around the world [4, 13].

With the advent of computer clouds, disaster recovery of data loss is now possible for the education sector. The traditional techniques used for disaster recovery are very expensive, and the education sector could not afford it due to limited funds [10]. Scientific documentation on free computational clouds is scarce, and it is even more limited on disaster recovery on a free computational cloud. The application of cloud computing in the educational field is at an early stage in the scientific literature [14].

III. Research design

This research focused on implementing a recovery plan in case of disasters by using free computational clouds. To achieve this, we applied the Research-Action methodology, framed in a bibliographical, descriptive analysis, and in a quantitative/qualitative evaluation. In particular, we analyzed several free computational clouds and their relation to the processes that contribute to execute educational programs, when diagnosing the natural environment where the earthquake of April 16, 2016 took place, at the Universidad Laica Eloy Alfaro de Manabí (ULEAM), Ecuador.

The quantitative data were collected from experimental tests performed on each EFSS, which culminated in the development of a routine Shell script under the methodology of Experimental Software Engineering, specifically based on evidence. The main purpose was to improve decision making regarding the development and maintenance of software, integrating the best current evidence of research with practical experiences and human values [15].

To evaluate the EFSS, we carried out two activities. First, we evaluated and implemented three open source options: Nextcloud, Pydio and Seafile. Second, we prepared two test servers with the following characteristics: Intel (R) Xeon (R) CPU E3-1220 v3 @ 3.10GHz with 4 cores; 4 GB RAM memory; Ubuntu Server 14.04 with a kernel 3.13.0-85-64-GNU/Linux Ubuntu; two hard drives of 1 terabyte each. Each EFSS presents a logical layer architecture (Fig. 1), where the client layer is the interface or front-end of the user, with the services offered by the application. Ubiquity is a feature present in EFSS, which allows access from anywhere and at any time.

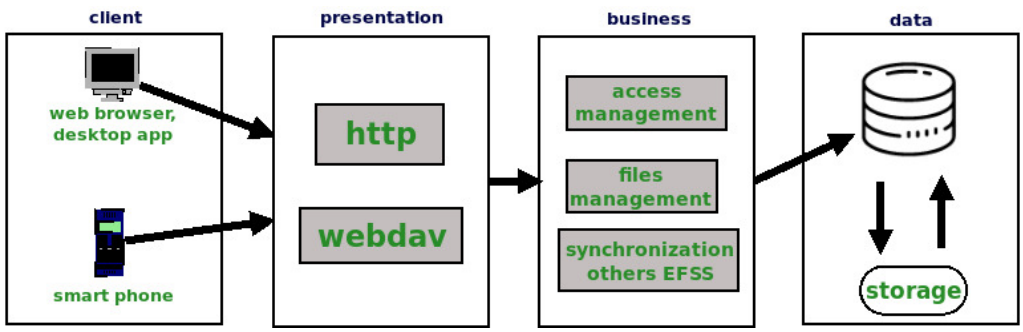

Fig. 1. EFSS logical layers.

IV. Quantitative and qualitative evaluation of the EFSS

According to a thorough review of the state of the art, there are more than 100 commercial EFSS solutions in the market [16]. However, due to the nature of this research, in which the University that served as a diagnosis is public, we framed the analysis in EFSS of free licensing.

Table 1 compares the characteristics of the EFSS, in terms of synchronization and storage. The installation requirements of each EFSS focused on an Apache web server, and the compatibility with databases such as MySQL and the GNU/Linux operating system. The versions of the EFSS analyzed were (a) Nextcloud 11.0.3, (b) Seafile 6.0.9, and (c) Pydio 7.0.4; some of the EFSS depend on packages according to the programming language in which they were developed. After implementing the EFSS, we installed the monitoring applications to obtain accurate data for the evaluation: (a) JMeter, an application written in Java open source, tests the performance and functional behavior of Web applications; we used the 2.13.20 version for GNU/Linux [17]; (b) Cacti, the front-end for RRDTool, stores data that comes from the RRDTool database and shows them graphically, as

well as stores the data in a MySQL database; we used the 0.8.8 version [18]; and (c) Mrtg, the front end for Snmpd (Simple Network Management Protocol), which is a daemon that collects information from the computer to make it available to other users; we used the 2.17.4 version [18].

Table 1
EFSS characteristics: Synchronization and Storage

	Characteristic	Nextcloud	Seafile	Pydio
Synchronization	Portable	√	X	√
	Conflict detection	√	√	√
	Rename and move files	√	√	√
	Version control	√	√	√
Storage	File protection with additional key	√	X	√
	Security / Encryption	ISO/IEC 270001:2013	HTTPS/TLS	Multiple-factor authentication
	Federation Sharing / Integration	√	X	X
	Additional features	A real-time collaborative tool; Calendar Contacts; Audio-video streaming	X (Enterprise)	X (Enterprise)

To carry out the tests, we assigned three (3) user segments of 1000, 6000, and 15000 threads each, which simulate 1000, 6000, and 15000 accesses of concurrent users, respectively. We chose these numbers based on the universe of students enrolled in the ULEAM [19]. JMeter, which is a loading tool to carry out simulations on any software resource, was configured so that the requests were completed by downloading a file with a size of 500 Kb.

V. Experimentation with EFSS in the recovery environment in case of disaster

After evaluating the EFFS, we implemented a synchronization process from the main to the secondary server; in Fig. 2 the broken line from the user to the current server represents the transparency that the user will have when switching from primary to secondary. We focused the tasks on synchronizing the time of the two servers, using as reference the Network Time Protocol (NTP) service of the Oceanographic Institute of the Ecuadorian Navy.

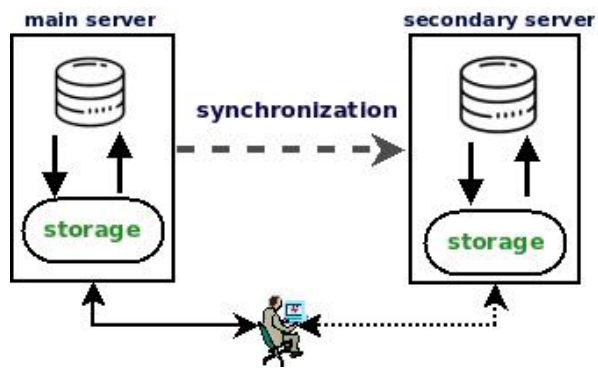

Fig. 2. Desired Architecture.

Subsequently, an SQL file from the database called *dbnc* was generated in the main server, as well as the file *db.md5* in order to check for integrity. We added an entry to the Linux *crontab* file, so it would execute every 10 minutes.

On the secondary server, we run another routine to verify whether the http service of the main server was operating or had failed, and to take the appropriate measures, checking integrity, transferring data, sending alert messages, and raising the http service. Finally, we added an entry in the *crontab* file of the secondary server, so it would execute every 10 minutes (Fig. 3).

```
#!/bin/bash
#Script que verifica que el estado del servicio del servidor principal
#y levanta el secundario, ademas se asegura de tener una ultima
#copia de base de datos.
# En los comentarios de este script se han eliminado
# intencionalmente las tildes y los caracteres especiales
# Elaborado por Ana Isabel Delgado Dominguez
FECHA=$(date +%d%m%Y-%H%M)
cd /var/www/html/nextcloud/
rm -rf dbnc.sql
rm -rf dbnc_comparar.md5
rm -rf dbnc.md5
rsync -apvr root@a.b.c.d:/opt/scripts/dbnc.* .
md5sum dbnc.sql > dbnc_comparar.md5
md5_comparar=$(cat dbnc_comparar.md5)
md5=$(cat dbnc.md5)
if [ "$md5_comparar" = "$md5" ];
then
   rsync -apvr root@a.b.c.d:/var/www/html/nextcloud/* .
   #Verificar estado del servicio http del servidor principal
   ESTADO=$(nmap 67.205.112.196 -p 80 | grep -o open)
   echo $ESTADO
   if [ "$ESTADO" != "open" ];
   then
        echo "servidor entra en produccion"
        #Se debe implementar funcion de envio de correo de notificacion
        echo "Servidor secundario ingresa a produccion" | mail -s "Alerta
NextCloud" anadelgado@gmail.com -r "administrador"
        mysql --user=userncdb --password=administrador ncdb <
ncdb.sql
        service apache2 start
    else
         service apache2 stop
    fi
else
    echo "Error al verificar integridad de transferencia verificar lo más
pronto posible" | mail -s "Alerta NextCloud" anadelgado@gmail.com
-r "administrador"
fi
```

Fig. 3. Routine Shell script for the recovery of an EFSS, in case of disaster.

Disaster recovery plan incorporating the routine (fragment)

To include the development of the routine within a disaster recovery plan for EFSS, we worked based on the ISO/IEC 22301 [20] standard, which focuses on Business Continuity Systems. From the gathering of information to the Central Computer Coordination Unit of the ULEAM, the planning to mitigate and prevent disasters on the services around the EFSS was completed in detail. The Government of Alberta (Canada) developed the template [21].

A. Authorization for the disaster recovery plan}

To elaborate this plan, we informed the highest authority of the Central Computer Coordination Unit.

Policies: Due to the lack of policies in the Central Computer Coordination Unit of the ULEAM before the earthquake of April 16, 2016, each department had its own Internet provider, its own IT staff, and an independent infrastructure depending on the area. However, the effects of the earthquake were evident, not only in material losses, but also in school desertion.

B. Scope of the disaster recovery plan

Table 2 describes the critical services in the EFSS.

Table 2
Critical services in the Efss

Level of Service	IT Service or Application Name	Recovery Time Objective, (RTO)	Recovery Point Objective, (RPO)
0	Authentication service	10 min	0
0	Download and upload service	10 min	0
0	File sharing service	10 min	0
1	Federation service with other EFSS	10 min	60 min
0	Instance synchronization service	10 min	0

- Assumptions

The Disaster Recovery Plan intends to provide the authorities with the information necessary to resume the EFSS service in an appropriate and timely manner, for the following scenarios:

- EFSS server hard drive failures
- Power failures of the data center or the servers cold room
- Corrupt database
- No Apache service (HTTP)
- Total disqualification of the data center, due to any type of disaster
- During the failure of the previous points, the secondary server located outside the institution will be enabled, either in a national or international data center.

The recovery procedures and the estimated time for the RTO (Target Recovery Time) and the RPO (Time Recovery Point) are based on assumptions that need validation:

- Implementation of a secondary server outside the institution with physical characteristics like those of the main server and equal software configurations
- Transfer of new files every 10 minutes from the main server to the secondary server
- Backup and verification of the integrity of the database
- The generated script for the transfer and treatment of the database generates alerts that should be taken seriously.
- Test period to make sure that the solution remains in operation.

C. Test plan

- Roles and responsibilities

The Disaster Recovery team, taking into consideration the current IT organization chart, will conform as follows (Fig. 4):

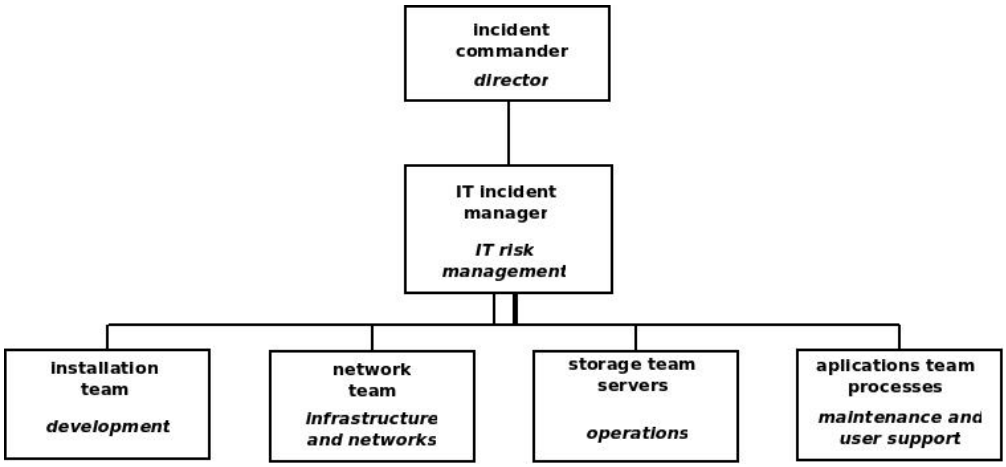

Fig. 4. Adapted IT organization chart.

The order of the individual tests performed before and after the interruption of the service must be the following:

- DNS and response times tests
- Integrity of database
- Integrity of data
- Apache services, MySQL without new features at startup
- User authentication service
- File download and upload service
- Directory and file sharing services

B. Verification of usability

In order to measure Usability, this research was based on the ISO/IEC 25000 series [22], also known as SQuaRE (System and Software Quality Requirements and Evaluation), which aims to help developing and acquiring products that fulfill the quality requirements and their evaluation [23]. Subsequently, we defined metrics of quality of use and weighted according to the studied reality. The evaluations of these metrics started from the definition of the evaluator (independent technician) and the scenario of the control tests in the main and secondary servers.

VI. Results and discussion

A. Results of EFSS

The JMeter, Cacti, and Mrtg tools evaluated the performance of each implemented EFSS, to obtain concrete results on homogeneous samples. Subsequently, the EFSS that generated the best results was subjected to the routine Shell script developed within the test servers. The routine monitored the main instance and executed backups in the secondary instance.

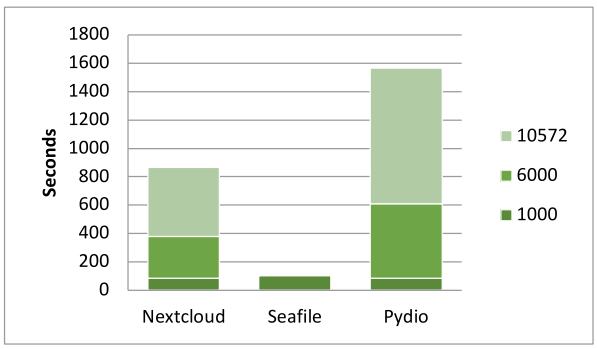

Fig. 5. Time for all samples.

The results focused on three elements: (a) the total and mean (μ) performance time used for all samples requested with a certain number of concurrent users; (b) concurrent users supported by the EFSS before collapsing; and (c) performance aspects in the server processor. In this way, Figure 5 shows the mean (μ) time (in seconds) that the EFSS took to execute the set of assigned requests; Figure 6 shows the number

of concurrent users that the EFSS supported when executing the assigned requests; Figure 7 depicts the average CPU consumption of each EFSS after 1, 5 and 15 minutes of execution; and Figure 8 details the average consumption of RAM memory for the execution of 6000 assigned requests.

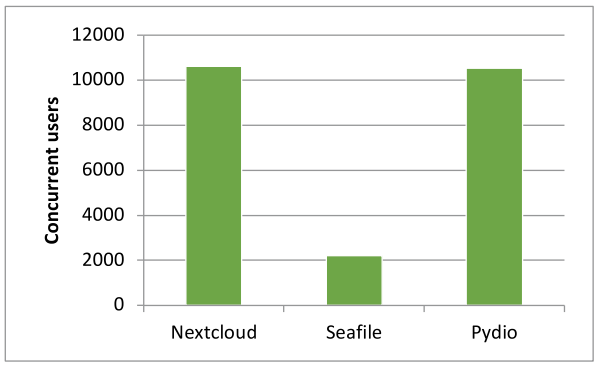

Fig. 6. Tolerance of user concurrence.

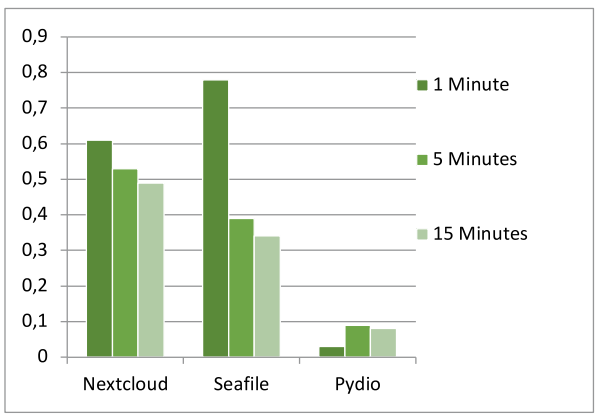

Fig. 7. CPU consumption.

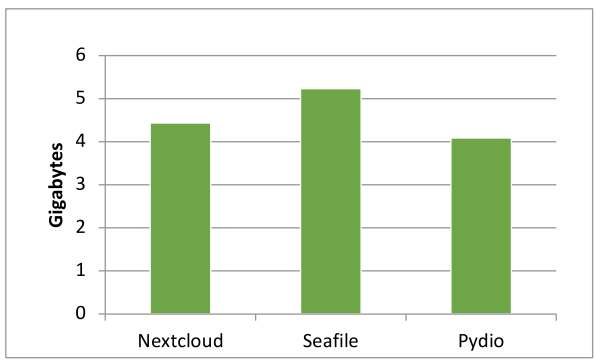

Fig. 8. Memory consumption.

Nextcloud was the EFSS with the best response rate, with a mean (μ) response time per request of 60 ms. Likewise, the number of defined users demonstrates the EFSS behavior for each group. Nextcloud maintained its response time, while Pydio increased considerably its response times as the stress test ran. Regarding the tolerance of concurrent users, Seafile was the first EFSS that did not support the stress test, with a number inferior to 2300 concurrent users downloading a 500 Kb file. In contrast, Nextcloud had a greater tolerance with the load of 10603 concurrent requests, maintaining the response time in all the scenarios. Finally, Pydio responded to no more than 10500 concurrent users with an equitable response time for each user group.

B. Usability of the routine

When selecting metrics on the Quality of Use, we used log monitoring applications on the main processes of the routine. The results showed that the developed routine meets the requirements and generates a high degree of satisfaction (Table 3). These results come from the following metrics: (a) Effectiveness; (b) Efficiency; (c) Satisfaction, and (d) Freedom of risk.

Table 3
Valuation of the usability of the routine

Quality	Quality of the routine	Score level	Degree of satisfaction
Use	9,00	Meets the requirements	Very satisfying

C. Discussion

The data indicate that Nextcloud performed remarkably better than its competitors, which have restricted certain modules and have a technical team under a business structure, instead of community and educational features such as those found in Nextcloud. On the other hand, the lack of a module that allows synchronizing the instances in Nextcloud was notorious. This motivated the implementation of a routine that added this feature, which we later verified in terms of functionality and usability, yielding very satisfactory data (Table 3).

In general, the test plan contemplated three clearly differentiated milestones. The initial phase corresponded to the preparation of the necessary environment, and the implementation and configuration of the necessary tools. The next phase

was the execution of the control tests, following the designed plan. The final phase was the data analysis, in which all the data obtained during the previous executions were studied to present them in such a way that they provide as much information as possible. From there, we proceeded to develop a routine to synchronize and remove the backup instance, product of the absence of this functionality in Nextcloud. The concept tests were carried out and the usability was evaluated. As a final product, the Disaster Recovery Plan was elaborated.

The opportunities found in Nextcloud are based on the nature of the project, which is completely open source, allowing us to detect a growth opportunity and thus develop a Bash routine to offer the continuity of storage and synchronization services. However, this implies a future development, which allows incorporating an emergency module and/or continuity in the configuration panel of the EFSS.

VII. Conclusions and Future Work

In the present study, we configured three computational clouds EFSS under free licensing, and evaluated them quantitatively by measuring the time they used to respond all the requests and each individual request, as well as the number of concurrent users supported and the consumption of CPU and RAM resources. The results of these tests determined that Nextcloud is the best EFSS to implement in an educational scenario, taking into account the impact it generates and its real-time collaboration features. Subsequently, and given the absence of a synchronization functionality of Nextcloud instances, it was necessary to develop a routine that allows business continuity in the face of a disaster. This routine and subsequent Disaster Recovery Plan were prepared based on the ISO/IEC 25000 and 22301 standards. As future work, we plan to include the routine as a back-end package within the EFSS, for its implementation in GNU/Linux distributions.

Authors' contributions

Delgado-Domínguez conducted the search, the recompilation, and the analysis of the papers referenced in this article, and contributed to write the manuscript. Fuertes-Díaz was the advisor of the project, and supervised the development of the DRP. Sánchez-Gordón contributed to write the manuscript and reviewed it. All authors read and approved the final manuscript.

References

[1] F. Daryabar, A. Dehghantanha, and K.-K. R. Choo, "Cloud storage forensics: MEGA as a case study," *Aust. J. Forensic Sci.*, pp. 1–14, Apr. 2016. http://doi.org/10.1080/00450618.2016.1153714.

[2] L. Yang, "Education Cloud: New Development of Informationization Education in China," in *Frontier and Future Development of Information Technology in Medicine and Education*, 2014. DOI: http://doi.org/10.1007/978-94-007-7618-0_131.

[3] S. Kamada, T. Ichimura, T. Shigeyasu, and Y. Takemoto, "Registration system of cloud campus by using android smart tablet," *Springerplus*, vol. 3 (1), pp. 761–774, 2014. DOI: http://doi.org/10.1186/2193-1801-3-761.

[4] T. Dong, Y. Ma, and L. Liu, "The Application of Cloud Computing in Universities' Education Information Resources Management," in *International Conference on Information Engineering*, 2012. DOI: http://doi.org/10.1007/978-1-4471-2386-6_122.

[5] Chrysikos and R. Ward, "Cloud Computing Within Higher Education: Applying Knowledge as a Service (KaaS)," in *Continued Rise of the Cloud: Advances and Trends in Cloud Computing, Z*, 2014. DOI: http://doi.org/10.1007/978-1-4471-6452-4_13.

[6] D. C. Wyld, and R. L. Juban, "Education in the Clouds: How Colleges and Universities are Leveraging Cloud Computing," in *Technological Developments in Networking, Education and Automation*, 2010. DOI: http://doi.org/10.1007/978-90-481-9151-2_1.

[7] J. Zhang, S. Sagar, and E. Shihab, "The Evolution of Mobile Apps: An Exploratory Study," in *International Workshop on Software Development Lifecycle for Mobile*, 2013. DOI: http://doi.org/10.1145/2501553.2501554.

[8] J. C. Augusto, "Ambient intelligence: opportunities and consequences of its use in smart Classrooms," Innov. Teach. Learn. Inf. Comput. Sci., vol. 8, no. 2, pp. 53–63, Jun. 2009. DOI: http://doi.org/10.11120/ital.2009.08020053.

[9] Y. Shi, H. H. Yang, Z. Yang, and D. Wu, "Trends of Cloud Computing in Education," in *7th International Conference Hybrid Learning*, 2014. DOI: http://doi.org/10.1007/978-3-319-08961-4_12.

[10] K. B. Nayar, and V. Kumar, "Benefits of Cloud Computing in Education During Disaster," in *International Conference on Transformations in Engineering Education: ICTIEE*, 2014. DOI: http://doi.org/10.1007/978-81-322-1931-6_24.

[11] Gregus, and V. Karovic, "Practical Implementation of Private Cloud Based on Open Source ownCloud for Small Teams - Case Study," 2016, pp. 183–187.

[12] I. Arpaci, "Antecedents and consequences of cloud computing adoption in education to achieve knowledge management," *Comput. Human Behav.*, vol. 70, pp. 382–390, May. 2017. DOI: http://doi.org/10.1016/j.chb.2017.01.024.

[13] V. Paranjape, and V. Pandey, "An Innovation in Education Through Cloud Computing," in *All India Seminar on Biomedical Engineering*, 2012.

[14] X. Wang, and Q. Cai, "The Analysis of the Application of Cloud Computing in the Field of Basic Education," in *Second International Conference Technology in Education*, 2015. DOI: http://doi.org/10.1007/978-3-662-48978-9_16.

[15] S. Pizard, F. Aceranza, V. Casella, S. Moreno, and D. Vallespir, "Conceptos de Ingeniería de Software Basada en Evidencias," *Reporte Técnico*, RT 15-08, 2015.

[16] M. Basso, C. Smulders, and J. Mann, "Magic Quadrant for Enterprise File Synchronization and Sharing Market," *Gart. Inc.,* Jul. 2015, pp. 16, 2014.

[17] F. J. Díaz, C. M. Banchoff Tzancoff, A. S. Rodríguez, and V. Soria, "Usando Jmeter para pruebas de rendimiento," in *XIV Congreso Argentino de Ciencias de la Computación*, 2008.

[18] S. Powers, "Graph Any Data with Cacti!," *Linux Journal*, vol. 271. pp. 50–68, Nov. 2016.

[19] ULEAM, "Reporte de matriculados 2016 y 2017," Universidad Laica Eloy Alfaro de Manabí, 2017.

[20] ISO/IEC, "ISO/IEC 22301:2012 Societal security -- Business continuity management systems --- Requirements," 2012.

[21] Alberta Education, "IT Disaster Recovery Planning Guide", 2016. Available in: http://education.alberta.ca/media/3272747/2-it-disaster-recovery-planning-guide.pdf.

[22] ISO/IEC, "ISO/IEC 25010 (2011) - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models," 2011.

[23] K. Esaki, "Verification of Quality Requirement Method Based on the SQuaRE System Quality Model," *Am. J. Oper. Res.*, vol. 3 (1), pp. 70–79, 2013. DOI: http://doi.org/10.4236/ajor.2013.31006.