Scalable IT for e-Assessment

Bastian Küppers[1], Richard Zameitat[2], Thomas Eifert[3], Ulrik Schroeder[4]

[1]IT Center / Learning Technologies Research Group, RWTH Aachen University, Seffenter Weg 23, 52074 Aachen, kueppers@itc.rwth-aachen.de
[2]IT Center, RWTH Aachen University, Seffenter Weg 23, 52074 Aachen, zameitat@itc.rwth-aachen.de
[3]IT Center, RWTH Aachen University, Seffenter Weg 23, 52074 Aachen, eifert@itc.rwth-aachen.de
[4]Learning Technologies Research Group, RWTH Aachen University, Ahornstraße 55, 52074 Aachen, schroeder@cs.rwth-aachen.de

Keywords
e-assessment, electronic examinations, scalable IT

1. SUMMARY

Electronic assessments are a topic of growing interest for universities. Even though it is not a new approach, there is still no “common way” for implementing those assessments. This leads to a situation where universities start implementing a suitable approach themselves. However, it has to be ensured that such an approach scales to large numbers of students. This paper describes the technological necessities for implementing a software framework for electronic assessments that utilizes a Bring Your Own Device (BYOD) approach.

2. INTRODUCTION

Electronic assessment can be conducted in a computer lab of the educating institution. However, this approach has some drawbacks: high costs and a lot of management overhead are introduced (Biella, 2009)(Bücking, 2010). However, the biggest drawback arises for the students, as they have to work on unfamiliar devices. If some students have to work in an unfamiliar environment, this is unfair in comparison to students who do not have this handicap. Utilizing a BYOD approach, i.e. using the students’ own devices in the examination, is a solution to the aforementioned issues. However, a BYOD approach also introduces new problems. For example, the different capabilities of the students’ devices have to be considered to prevent unfairness. A possible solution at this point is to implement an approach that offers the possibility to offload computationally intensive tasks to a server (Kovachev, 2017), therefore smoothing the differences between the students’ devices. A major issue introduced by this approach is *scalability*. The approach is not suitable for electronic assessments, if students have to wait too long for the server’s response. However, this problem cannot be solved by simply adding more processing power, as potential bottlenecks in the process have to be identified to make the solution scalable.

This paper describes what has to be considered to make such an approach scale to large numbers of students. The focus is put on the capabilities of hardware, and software engineering decisions and how those affect the performance of a scalable software solution. A case study on a JAVA build server that also offers remote debugging capabilities is discussed and the lessons learnt from this project are presented. Additionally, a field test with about hundred students and the differences of performance in comparison to a mock-up lab test are discussed.

3. THE FLEX PROJECT

The /FLEX/ project (Framework For FLExible Electronic EXaminations) was initiated to provide solutions in the field of electronic assessment on students' devices. To achieve this, the goal was to create a platform that allows students to take electronic examinations using their own devices. To allow for nearly universal use, /FLEXclient/ was designed as an OS-independent application. Right from the start binaries were not only built for Microsoft® Windows and Apple’s MacOS, but also for

Linux-based systems. Based on web technologies for core functionality and exam modules, the potential for other usage scenarios, like apps for mobile devices such as tablets, was kept open. Minimal native code is bundle with releases of /FLEXclient/ to allow for platform specific integrity and security checks. Concerning security, and cheating prevention in particular, a so called "log-down" approach was chosen, instead of commonly applied lockdown. Here, the client application mostly monitors its own state and events, being less invasive towards the host system, but not significantly less secure compared to other solutions. The goal is to provide practical security, at least comparable to paper-based examination, not ultimate theoretical security. Figure 1 shows the architecture of the /FLEXclient/, highlighting the use of web technologies.

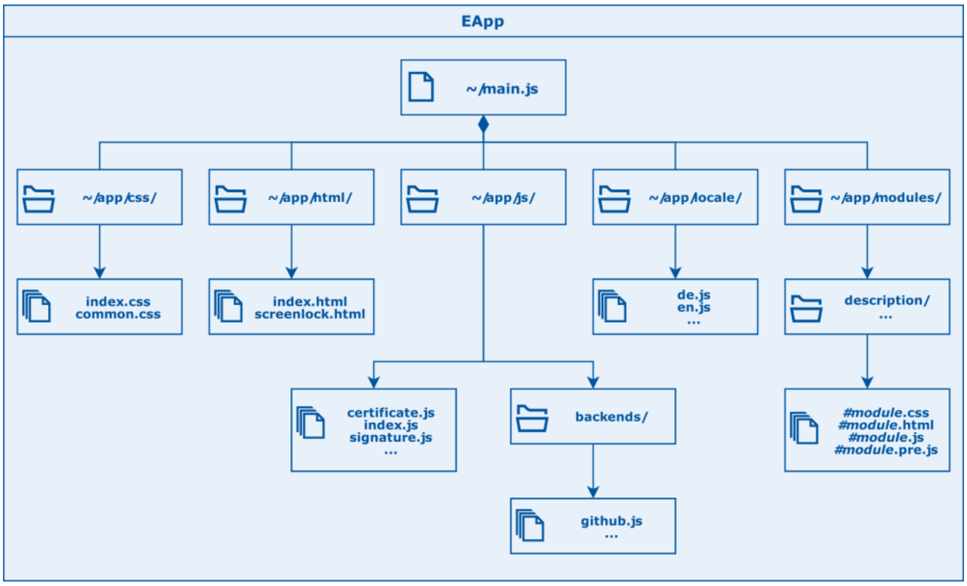

Figure 1: Architecture of /FLEXclient/

On the server side, /FLEXserver/, which consists of multiple micro services, builds on a hierarchical structure of microservices, interacting mostly via REST-APIs. Here, every microservice is only responsible for a specific task and does not depend on higher level services. On the top of this hierarchy is /FLEXproxy/ which accepts requests, uses authorization and exam registration services to validate their eligibility and forwards them to the responsible lower level services. This architecture allows for redundant services, scaling for higher demand and providing a fallback in case of failures, and easy development of new services, even for varying technologies. Additionally, resources for computational offloading were included into the architecture to be able to smooth out expectable differences between the students' devices. Figure 2 shows the architecture of /FLEXserver/.

4. EXPERIMENTS

For a smooth conduction of electronic examinations, it is important that /FLEXserver/ performs well under load in order to serve the students' requests during the exam in a timely manner. To verify the dependency on specific hardware and how /FLEXserver/ performs under load, several experiments were carried out.

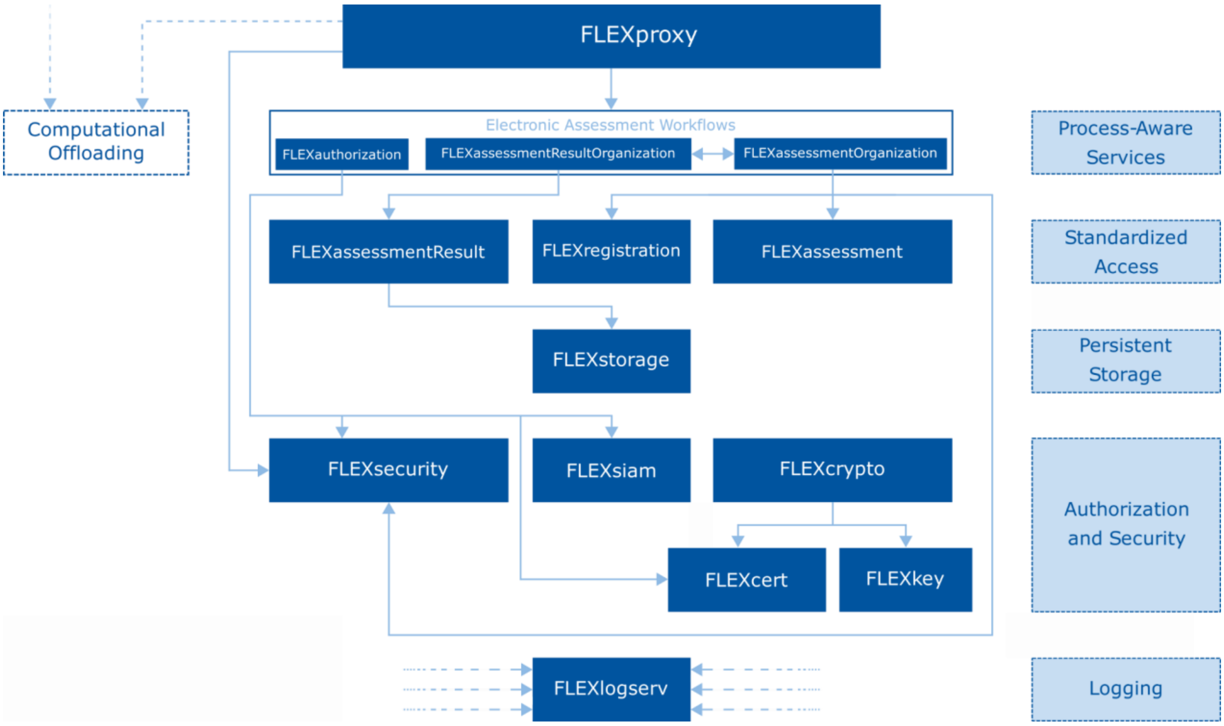

Figure 2: Architecture of the /FLEXserver/

4.1. THE OLD HARDWARE

The first version of /FLEXserver/ was executed on a virtual server with 4 virtual CPU cores and 8 GB of RAM. It was expectable that this set up would not scale to hundred or more students. However, these experiments were used as a base line for the interpretation of the performance of other set-ups. For the experiments, a certain amount requests to the server was made with a specified time interval between the requests. For example, 100 requests that are started every 2 seconds. For the test, the time for uploading a piece of source code, compiling it and executing the resulting executable in debug mode was measured.

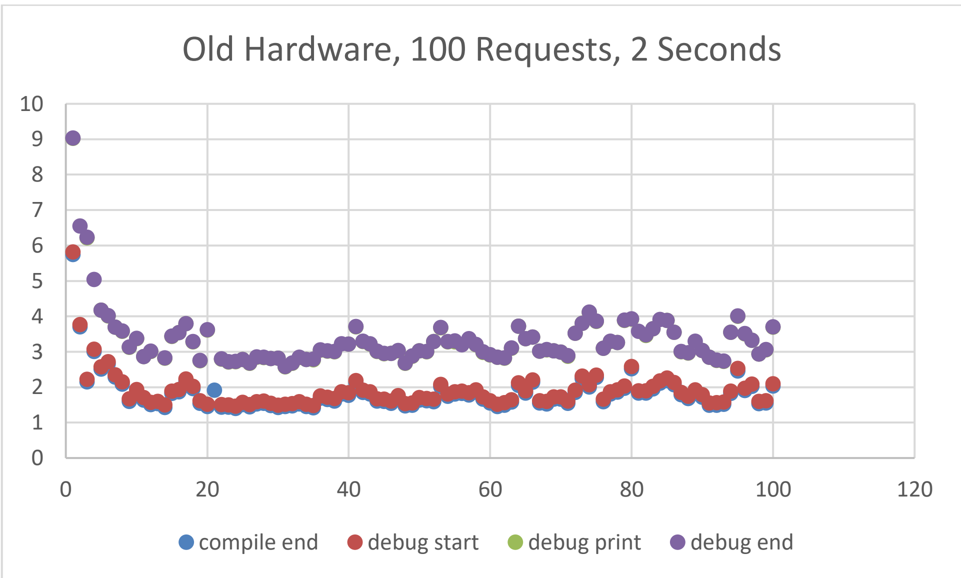

Figure 3: Old Hardware, 100 Requests, 2 Seconds

After an initialization phase, /FLEXserver/ is able to serve every request within 4 seconds, i.e. the time between the student's button press and the end of the debug process.

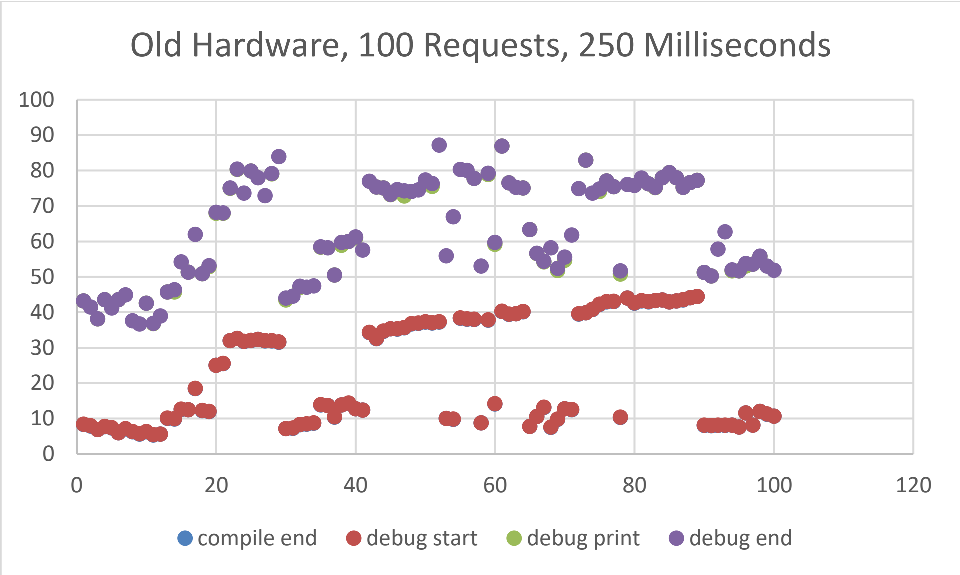

Figure 4: Old Hardware, 100 Requests, 250 Milliseconds

If the interval between each request is lowered to 250 Milliseconds, /FLEXserver/ is no longer able to serve the students requests in a timely manner. Students would have to wait for up to 90 seconds until /FLEXserver/ provides the requested result. Some of the requests did not even terminate properly anymore due to the overload on the server. With a growing number of students, however, it is not far-fetched that requests to /FLEXserver/ are made in similar intervals as have been tested.

4.2. THE NEW HARDWARE

Due to the disillusioning results of the experiments on the old hardware, an upgrade of the hardware was made. The new server is equipped with 32 CPU cores and 32 GB of RAM.

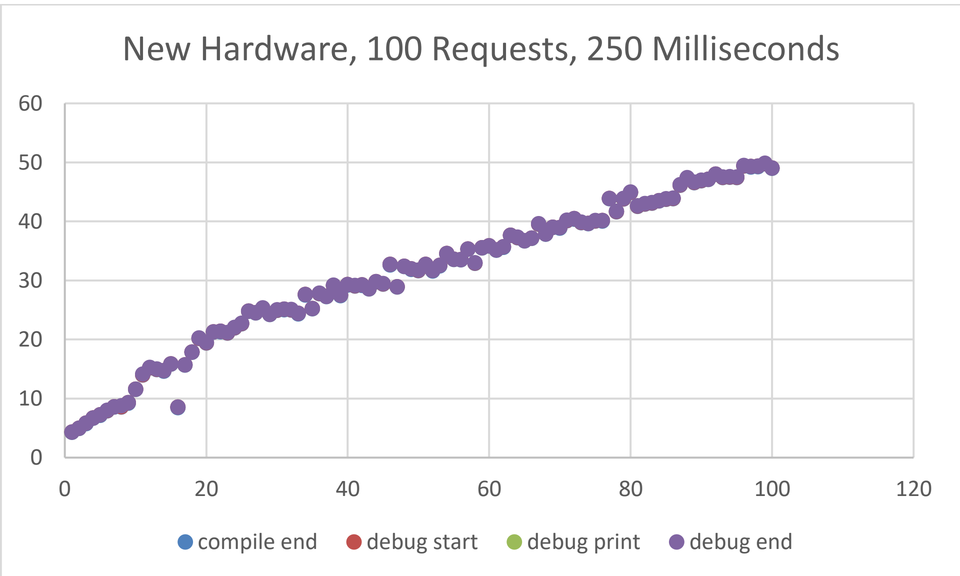

Figure 5: New Hardware, 100 Requests, 250 Milliseconds

Testing again 100 requests every 250 milliseconds, the new hardware performs much better than the old hardware. Still, the progression of the performance is not optimal. Admittedly, the performance is better than on the old hardware. However, the time for serving a request grows linearly with the number of requests.

5. DISCUSSION

The results clearly show that only increasing the computing capabilities of the server hardware does not solve the problem of scaling an application to large number of users. The old hardware had a much more chaotic behavior and a worst-case performance of up to 90 seconds. In comparison, the new server hardware has been upgraded by a factor of 8 in terms of the number of CPU cores and by a factor of 4 in terms of memory. Therefore, an increase in performance by a factor of at least 4 would be naively assumable. However, the worst performance for the same set of parameters yielded a worst performance of about 50 seconds on the new hardware – that is not even a factor of 2! Therefore, another factor seems to limit the performance.

In our scenario we were able to track down the bottleneck: the composition and launch of multiple docker containers, one per request, could not be handled as timely as necessary by the server. To overcome such a problem, the underlying hardware architecture has to be replaced. A single virtual machine is only capable of launching and handling so many docker containers due to the overhead introduced by resource allocation on the virtual machine. An alternative hardware architecture could be a cluster of multiple machines orchestrated by Kubernetes (Burns, 2016), which is especially designed to launch and handle a vast amount of docker containers in a timely manner.

6. SUMMARY AND FUTURE WORK

The paper presented first experimental results regarding the scalability of a software for electronic examinations. The results showed that simply improving the computational capabilities of the server hardware does not automatically guarantee that a software solution scales in the same way as the hardware has been upgraded.

The next steps include the setup of a Kubernetes cluster and the deployment of /FLEXserver/ on this new hardware environment. Further experiments will be carried out to verify that this new hardware setup will indeed improve the performance of /FLEXserver/.

7. REFERENCES

Biella, D., Engert, S. and Huth, D. (2009) Design and Delivery of an E-assessment Solution at the University of Duisburg-Essen. *Proceedings EUNIS 2009, ser. EUNIS Proceedings, vol. 15* [Online] https://www.unidue.de/imperia/md/content/zim/veranstaltungen/eunis_09_e-assessment.pdf

Bücking, J. (2010). eKlausuren im Testcenter der Universität Bremen. Ein Praxisbericht. Retrieved März 01, 2017, from E-Teaching.org: https://www.e-teaching.org/praxis/referenzbeispiele/eassessment-bremen

Kovachev, D., & Klamma, R. (2017). Framework for Computation Offloading in Mobile Cloud. *International Journal of Artificial Intelligence and Interactive Multimedia*, pp. 6-15.

Burns, B., Grant, B., Oppenheimer, D., Brewer, E., Wilkes, J. (2016). Borg, Omega and Kubernetes. *ACM Queue*, vol. 14, no. 1, pp. 70-93

8. AUTHORS' BIOGRAPHIES

Bastian Küppers, M.Sc. is research associate at the IT Center of RWTH Aachen University. His research focusses on e-Learning and e-Assessment technologies. He received his M.Sc. cum laude in Artificial Intelligence from Maastricht University in 2012. Since 2010 he works at the IT Center as a software developer and later as a teacher for parallel programming, robotics and other topics in computer science for the study program "Applied Mathematics and Computer Science" at FH Aachen University of Applied Sciences.

Richard Zameitat, B.Sc. received his B.Sc. in Scientific Programming at FH Aachen University of Applied Sciences in 2017. Since then he works as software developer at the IT Center of RWTH Aachen University.

Dr. rer. nat. Thomas Eifert is Chief Technology Officer of the IT Center of RWTH Aachen University and as such responsible for the strategy for technological development and the corresponding third-party funding of the IT Center. The focus in this function includes concepts for research-oriented storage infrastructures. He is also a lecturer for Calculus in the study program "Applied Mathematics and Computer Science" at the FH Aachen University of Applied Sciences, where he is involved in the processing and digitalization of traditional teaching content.

Prof. Dr-Ing. Ulrik Schroeder received his Diploma degree as well as his PhD in Computer Science from Technische Universität (TU) Darmstadt. Since 2002 he heads the Learning Technologies Research Group in the computer science department at RWTH Aachen University. His research interests include assessment and intelligent feedback with a focus on learning processes, Web 2.0 applications and social software in education, mobile Internet and learning, gender mainstreaming in education, and Computer Science didactics.