Signing made easy – hiding complexity of eSignature solutions in a black box

Janina Mincer-Daszkiewicz[1*] and Tadeusz Gąsior[2]
[1] University of Warsaw, Poland
[2] OPTeam, Poland
jmd@mimuw.edu.pl, TGasior@opteam.pl

Abstract

For many years Polish Higher Education Institutions (HEIs) have been actively incorporating digital solutions. Through the financial support of the state, as part of the Digital Poland program (carried out from 2014-2020), universities deployed student management systems and the Ministry of Science and Higher Education built central systems collecting data from HEIs. Changes in law introduced in 2019 in higher education, opened the way towards a fully electronic equivalent of the so-called student records folder.

In early 2020, the world faced the COVID-19 pandemic which accelerated the digitalization process advancing forward the transition to remote handling of student information. As a result, many provisions were introduced by the Ministry of Science and Higher Education, which sanctioned the replacement of paper documents with electronic ones, provided that their authentication, integrity, non-repudiation and confidentiality are preserved.

The amount and diversity of documents produced in HEIs is substantial. Many of them need to be signed: both for internal use (e.g. student records supporting study processes) and for external use (e.g. documents on student achievements required for further studies or employment). Furthermore, some of these documents travel across borders due to the increasing internationalization of higher education in Europe and beyond.

Polish HEIs range in size from 1,000 to almost 50,000 students and often share the same student management system. Therefore, they would benefit from easy-to-share, customizable, simple to install and use, low-cost solution for storing digital certificates, signing documents and validating their signatures.

The subject of this paper is the **eSignForStudy** project which addresses these needs. The objective of the project is to design and develop a highly configurable eSignature solution to be used in the Polish higher education area, interoperable with **Erasmus Without Paper Network** for cross-border digital document validation.

[*] ORCID 0000-0003-0672-5291

1 Introduction

In Polish Higher Education Institutions (HEIs) digital signatures are used in many scenarios, e.g. for securing transmission of data over the network, signing documents for internal use and authenticating official documents with qualified signatures. These institutions vary in size from small (less than a thousand students) to large (tens of thousands of students). The **MUCI** consortium [9] includes over 80 HEIs, representing more than 50% of students from the public sector. They all use the same *University Study-Oriented System* (**USOS**) [12]. Therefore, developing digital solutions for USOS provides tremendous economies of scale.

In March 2019, DG EAC of the European Commission introduced the roadmap for the digitalization of the Erasmus+ Programme. This roadmap outlines specific dates for the introduction of the *Erasmus Without Paper* (**EWP**) standards [1]. EWP is a platform for the secure exchange of student records between HEIs [7], mostly focused on supporting student mobility processes [6]. It is currently developed and maintained under the umbrella of the *European Digital Student Service Infrastructure* (**EDSSI**) project [3]. Some of these student records need to be digitally signed and validated.

However, currently there are no easy-to-share, configurable (customizable to the scale of the institution), simple to install and use, low-cost solutions for storing digital certificates, signing documents and validating the signatures.

The **eSignForStudy** project [4], undertaken by partners from Poland, Ireland, and Czech Republic, addresses these needs and aims to produce a solution which can be used both locally and cross borders.

The primary objective of the project is to design and develop a highly configurable eSignature solution to be used in higher education, deploy it in the Polish higher education area and validate it through the cross-border exchange of documents using a European-wide platform for the secure transfer of student data between HEIs.

The project started with an in depth requirement analysis carried out through on-line discussions and meetings with Polish HEIs. The main project requirements identified and an overview of the defined system software architecture are included in **Section 2**.

University of Warsaw (UW), one of the members of MUCI, integrated eSignForStudy with USOS, for testing and validation. Details of this integration together with some decisions made are outlined in **Section 3**.

Czech Technical University (CTU) has developed its own student information system (KOS), with a built in International Mobility Module. EWP will be used for the secure cross-border transfer of documents between KOS in CTU and USOS in UW.

KOS has its own solution for signing documents and validating signatures. This gives the opportunity for cross-check and cross-validation of the two tools, the one from KOS, and the one built during the project, for the mutual benefit.

The solution is designed to resemble a black box with optional/interchangeable components and open interfaces to enable interoperability. Its architecture is explained and some technical details are given in **Section 4**. Polish IT company OPTeam SA, one of the partners in the project, with a long history of cooperation with Polish HEIs, is leading the development.

Finally, **Section 5** gathers some final conclusions on the project outcomes.

2 eSignForStudy project

The eSignForStudy project aims to solve specific problems resulting from the general complexity and variety of eSignature solutions, which often become an obstacle hindering the implementation of these solutions, both in small institutions (due to the lack of technical support from local administrators) and in large institutions (due to the lack of solutions that scale well).

The solution addressing these needs should be seamlessly integrated within any ecosystem. Then, it would resemble a **black box** with optional/interchangeable components and open interfaces to enable interoperability. End users (e.g. staff of student office or International Relation Office) will send requests from the system supporting their business processes (like USOS with its various subsystems), transparently triggering digital signature procedures which, in turn, will be forwarded to the black box using a unified interface.

The black box itself will be integrated with other external systems needed to fully resolve the request, like *Qualified Certified Authority* (**QCA**) or *Trusted Timestamp Services* (**TSS**).

All the underlying operational procedures and technical details linked to the developed solution, like the types of managed certificates, media on which they are stored, security measures, software components used for validation etc. should be hidden not only to the end user but also to system administrators, easing the deployment in the host institution infrastructure. To achieve this, the eSignForStudy solution will be distributed in the form of preconfigured containers, while keeping customizability and scalability characteristics to fulfill with size and growing workload institution demands.

Figure 1 shows the integration of eSignForStudy (green rectangle in the middle) in the institution software infrastructure.

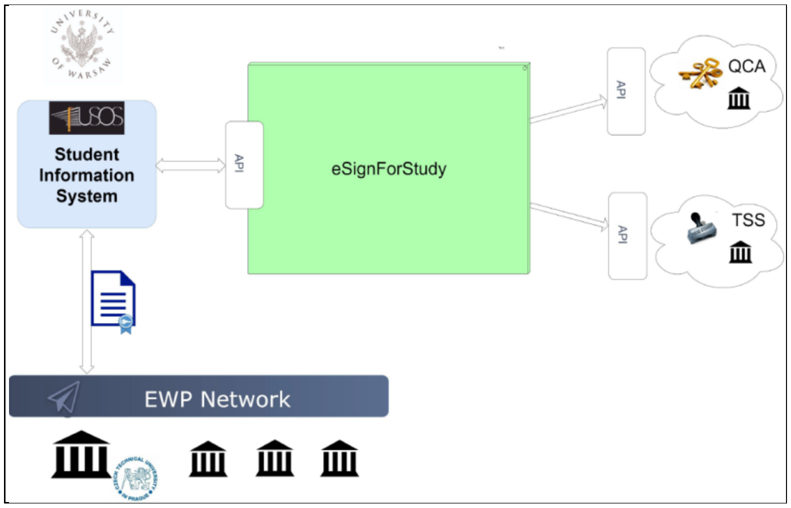

Figure 1: eSignForStudy – its place in the institutional software infrastructure

Although eSignForStudy is mainly focused to support educational requirements, it can also be configured and reused within any business sector, such as retail, finance, health care and beyond. Thus, the solution's internal architecture should not be biased by educational frameworks, but it should allow for operation in a stand-alone mode or as a subsystem available through dedicated API. A modular approach, where each module exports its own interoperable interfaces and handles different exchange protocols, is foreseen. This way the final solution will favor different business models, expanding OPTeam exploitation plan.

To sum up, the solution should address the following requirements:

- Supporting legal cross-border recognition of eSignatures, obtained from Poland and abroad.
- Support for various types of eSignatures, either qualified, or issued by internal Certified Authorities (CAs).

- Easy to use for end-users (no requirement to use cards/readers and other technical devices).
- Simple to manage, deploy and use in an environment where there is no highly qualified IT personnel (typical situation in higher education).
- Highly secure and highly efficient (where needed), e.g. supporting considerable signing of documents (e.g. thousands of student transcripts of records), using the institutional seal.
- Low cost.
- Easily exchangeable components to craft the solution to the needs of the institutions.
- Supporting set up of complex eSignature workflows and addressing the need to have documents signed more quickly.
- Supporting timestamps for long term documents.
- Supporting logging of transactions for audit purposes.

3 USOS integrated with eSignForStudy

University Study-Oriented System (USOS) is a student management information system used in many Polish educational institutions, which, thanks to the collaborative development approach, has evolved into a complete information and administration system for higher education.

In the next sections, we first identify some of USOS applications involved in document digital signing and validation. They all make use of the USOS SIGN specific application and the storage ecosystem deployed for USOS. Those will be also described, and will help in identifying the needs and requirements that eSignForStudy integration must fulfill to further enhance USOS capabilities.

3.1 USOS applications selected for pilotage

During the initial project stages as part of the business requirements analysis, we identified processes and supporting systems belonging to the USOS family, which involve digital signing. There are many, but after consultation with other HEIs using USOS, we have chosen the following processes for the pilot implementation:

- **IRK** (*Internetowa Rekrutacja Kandydatów*) is the admission portal. Applicants for studies get the administrative decision after the admission process is finished:
  - Polish citizens – all negative decisions have to be signed.
  - Non Polish citizens – all decisions have to be signed, positive and negative.
- **USOSadm** is the application for the administration. It contains the module for handling Erasmus+ mobility. *Transcripts of Records* (ToRs) are generated for incoming students after they complete courses, and then delivered to the student home university. ToR is created in the ELMO format with embedded PDF, which should/could be signed. Bilateral *Inter-institutional Agreements* (IIAs), signed between partners in mobility, can also carry embedded PDF, digitally signed.
- **USOSweb** is a web portal for students and academic teachers. Students and doctoral students apply for scholarships there. All issued decisions concerning financial aid are signed and presented to students for collection.

The analysis of the requirements revealed that the documents involved in the above mentioned processes, which will be stored in the institution’s repositories for the longer time period, need to be signed with the qualified timestamp. Another important conclusion was that it makes sense to sign documents stored in the electronic repositories even if qualified signatures are not necessary – to preserve their authentication, integrity, non-repudiation and confidentiality. However, these issues have not been tackled within the project and therefore they are not described in this paper.

3.2 USOS signing solution

Currently, amongst the various USOS applications where signing is involved, the one in charge of document signing is **USOS SIGN** [12]. The most tangible limitations of USOS SIGN, as compared with eSignForStudy, are the following:

- It is a desktop application that must be installed on the workstation of every employee who signs documents (for largest Polish HEIs these means hundreds of installations).
- Only one signature can be made per request.
- It does not support certificates stored in the cloud.
- It does not support certificates stored in Hardware Security Module (HSM).
- It does not support university seal.
- It does not support timestamps.
- It is not possible to verify signatures.
- It is not possible to add visual form of a signature on a signed document.
- No user interface is available, so a specific one has to be developed for every application using USOS SIGN.

USOS SIGN obtains the document to be signed from the browser running on the same workstation. It interacts with the secure device or token storing the certificate, which has to be attached to the same workstation. In some scenarios it also interacts with another device on which the signed object is stored (this is the case of a student identity card stored on the chip). It reports the carried steps in the log file and returns the signed document to the browser.

Due to the identified gaps in functionalities, as part of the eSignForStudy integration process, we plan to give universities the option to choose, at the application configuration stage, what tools they will use. Relevant configuration parameters will drive the process, forwarding signature requests to either a local USOS SIGN application or the remote eSignForStudy service.

3.3 Document storage

The key features of the eSignForStudy are signing and securely storing signed documents. The analysis of the requirements shows that the number of files to manage depends on the process. In this sense, signing operations can be applied to individual files or in batch where thousands of files are sent for signature in one request (e.g. in the order of 5,000). This means that the solution should also incorporate a transfer of documents between the system sending requests and the signing system.

Depending on the size of the HEI and the amount of files to handle, different installations of USOS use different database backend for storage. Small institutions that do not generate a large amount of data use Oracle Express Edition, which is free of charge but has some storage limitations. Large institutions, which create and store huge amount of documents, use commercial versions of Oracle, which do not impose any space limitations.

In order to support the needs of both small and large HEIs, USOS designers decided to harmonize the solution and keep large binary objects outside the main USOS Oracle database. A separate repository for such objects, known as **Blobbox**, was designed and implemented some years ago [11]. Blobbox is used to store signed reports with student achievements, files sent via the EMREX Network, ToRs and Learning Agreements sent via the EWP network, and many more types of binary objects.

Blobbox is supervised by USOS API, which means that any USOS application which wants to use an object from Blobbox needs to call the corresponding USOS API methods [11].

Considering Blobbox for integration of USOS with eSignForStudy seems to be a good option. However, USOS API, which is the main application service for many other processes running across various applications from USOS family, may become a bottleneck when serving as an entry point for the object storage on a massive scale.

We anticipate that in the academic settings the demand for high-quality and efficient document storage will grow. Furthermore, we also have to take into consideration that some universities, especially the large ones, may prefer to outsource storing binary objects by delegating this functionality to existing cloud solutions, such as Microsoft Azure or Amazon.

For this reason, we considered alternative solutions. Amongst the publicly available, open source, ready solutions for object storage, two were recommended: OpenStack Swift [10], and MinIO [8].

OpenStack Swift is a very powerful set of tools; however the effort to deploy it locally may be an obstacle for smaller HEIs. In addition, USOS developers are familiar with MinIO as it is used in the infrastructure as the storage backend for the cloud installation. MinIO is an easy to install solution and offers what we consider an extremely important feature, S3 API support. This means that if USOS will use S3 API to access object storage, eventually every HEI would be able to use any storage solution implementing S3 API (which is a commonly supported standard). MinIO also solves the authorization issues, as it includes features that allow access rights to be transferred or delegated at single object level.

Therefore, we selected MinIO as the storage repository for integration of USOS with eSignForStudy. The process of replacing Blobbox with MinIO will be carried on gradually. Initially, MinIO will only be used to store documents exchanged between USOS applications and eSignForStudy. In a second phase USOS will be updated to use MinIO and all Blobbox records will be migrated to the new storage.

3.4 Signing process in the integrated solution

Once eSignForStudy is integrated in the USOS workflow, the new signing procedure will be implemented. It is important to note that MinIO will act as the intermediary between USOS and eSignForStudy. In this sense, we have defined two bucket folders accessible from both applications, one storing incoming documents from USOS directed to eSignForStudy (**in folder**) and the other for documents in the opposite direction (**out folder**). The process comprises of the following steps (Figure 2).

1. A user prepares documents for signing in the USOS application.
2. The application stores the documents to the **in bucket's folder** in MinIO by calling the appropriate API. MinIO **in** and **out folders** are shared by USOS and eSignForStudy.
3. The application then sends a signing request to eSignForStudy. The request includes document identifiers in MinIO, and details of the operation, such as what type of signature is required, who signs it, whether a time stamp is needed, whether and where a visual form of the signature should be added to the document, etc.
4. The user is redirected to eSignForStudy which handles the whole signing process. This includes, in a loop, for each requested document:
   - retrieving the document from the **in folder** in MinIO,
   - signing the document,
   - storing the document in the **out folder** in MinIO.

5. Upon signing process finalization, the user is redirected back to the USOS application. The application collects the documents from the **out folder** in MinIO and stores them in their final location in the USOS database.

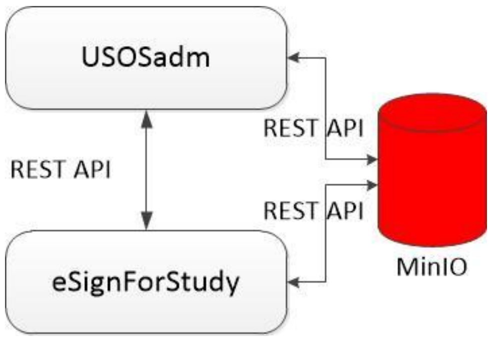

Figure 2: Communication between USOSadm and eSignForStudy using REST API and MinIO

The current implementation of the new workflow is not yet fully robust and additional work must be carried out to solve some issues that could appear during the signing process. For example, it may end with an error at any moment, which means that eSignForStudy has to properly report to the USOS application the status of the signing process for each document. Other issues might arise if the end user at any time opens the browser and enters the page where the process started. In such cases, the USOS application being aware that the signing is in progress should not allow the user to interfere with it.

4 eSignForStudy

4.1 Overall architecture of eSignForStudy

The general architecture of the final solution incorporated into the ecosystem of the University of Warsaw is depicted in Figure 3. Internally it supports the handling of certificates stored in local secure repositories – *Hardware Security Module* (**HSM**) or its software equivalent (**SSM**). If qualified signatures are needed and cannot be downloaded to the local infrastructure, they can be stored in a remote cloud managed by the *Qualified Certified Authority* (**QCA**) and handled from inside the institution's infrastructure by means of an API. The solution also enables the use of *Trusted Timestamp Services* (**TSS**) delivered by certified providers.

eSignForStudy eSignature Service is based on **DSS** open-source library [1] (by incorporating the library's code). It provides the means to create and validate electronic signatures and electronic seals compliant with eIDAS and related standards, to be used by Higher Education Institutions and private sector from member states of the European Union.

eSignForStudy solution offers two deployment alternatives: as a modular monolith solution using WildFly Java server (dedicated for small HEIs) or as containerized microservices dynamically managed according to a current system load handled by Kubernetes (for systems using multiple servers). Both of them are provided with the according guidelines for easy deployment and configuration.

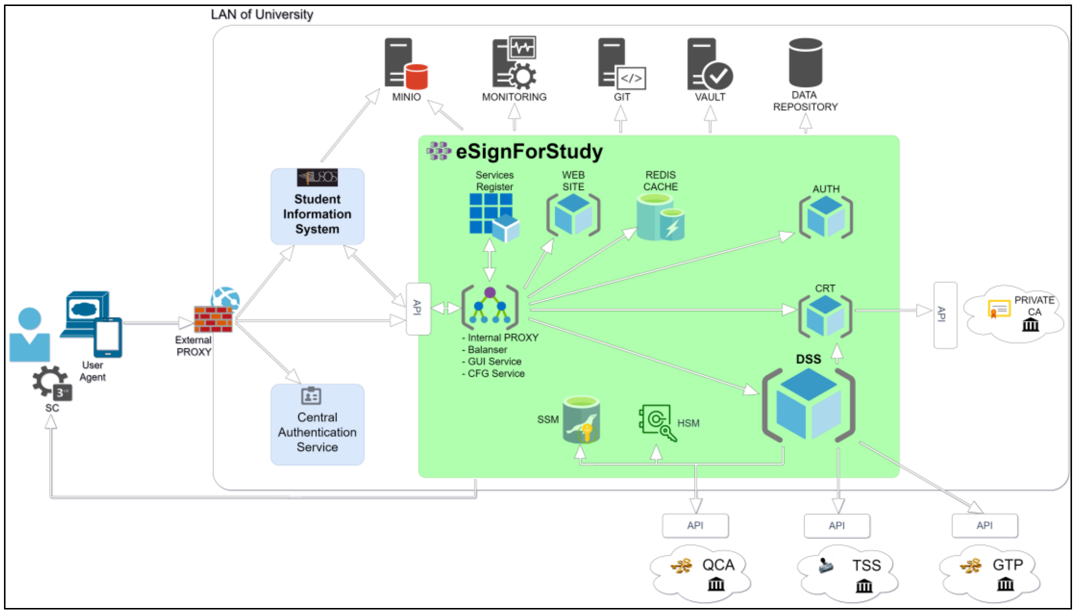

Figure 3: eSignForStudy – general overview of the internal structure

4.2 Functional high-level architecture overview

From a technical point of view, the main functionalities of the eSignForStudy are an advanced electronic signing with the use of a private key stored in various types of secure media (i.e. smart cards, HSM, or in the cloud) and the verification of the electronic signature attached to the document (using DSS libraries). In addition to these basic functions, the application aims to ensure data security, provide high availability and the possibility to quickly react to any failures.

Mentioned goals are achieved thanks to a suitable architecture and additional ***co-systems*** depicted in Figure 3 and listed in Table 1 (***External*** systems are located outside the institution network, while ***Shared*** systems work inside the network as standalone servers, eSignForStudy and USOS are skipped).

Software System	Description
Monitoring	***Shared***. Monitoring system of the institution network. (e.g. Sentry, Prometheus, Grafana, Telegraf).
GIT	***Shared***. Versioning system for configuration files.
VAULT	***Shared***. Management of tokens, passwords, certificates, encryption keys for protecting secrets and other sensitive data.
Data Repository	***Shared***. Relational database system (MariaDb | MySQL | Postgres | MS SQL | Oracle)
Private Certification Authority	***Shared***. Certification authority of the institution PKI infrastructure (EJBCA).
Hardware Security Module (HSM)	Users' private key store based on a hardware cryptographic module.
QCA	***External***. Qualified Certified Authority.
TSS	***External***. Trusted Timestamp Services.
GTP	***External***. Governmental Trusted Profile.
SC	***External***. Smart Card System on user workstation.

Table 1: Systems cooperating with eSignForStudy

Functionality of eSignForStudy is provided by the dedicated ***modules*** – implemented and deployed as microservices, each module delivered in a separate Docker container. Table 2 enumerates these modules and also describes their role within the whole ecosystem and the development technologies used. They are also depicted in Figure 3.

eSignForStudy Module	Description of provided functionality	Technology
Software Security Module (SSM)	An encrypted storage of user's private keys based on a database engine.	Java, MariaDb and PKCS#12
Hardware Security Module (HSM) Service	Service used by DSS to access HSM. It is needed for licensing reasons.	Java, PKCS#11
DSS	Online signing and signature verification via JSON/ HTTPS API. The main module of eSignForStudy, many instances of it will operate under high loads.	Java and Spring MVC
CRT	Online certificates managing and remote encryption.	Java and Spring MVC
AUTH	OpenId Connect, JWT tokens, OAUTH2 and OTP.	KeyCloak Application and extensions in JAVA
Application registry	Registration of microservices.	CONSUL
WEB SITE	Web pages.	Java and Spring MVC
Single-Page application	eSignForStudy functionality for end users via web browser.	JavaScript and Angular
Proxy and balance	Balances servers load.	NGINX
CFG	Online configuration for other modules.	Java and Spring MVC
AUDIT	Audit log.	Java and Spring MVC

Table 2: Modules of eSignForStudy and used technologies

eSignForStudy exposes a set of interfaces in order to integrate it with any external application as for instance USOS. It extends the original REST API implemented in DSS libraries [1] and aims to simplify the communication between applications when signing or verifying signatures for a series of documents. The main goal of the defined API is to implement the communication scenario designed for the project as described in section 3.4.

Three protocols have been designed to support different means for submitting documents for signing or validation, and uploading of signed documents or signature verification reports [3]:

- Documents are collected in the MinIO bucket in folder and send back via bucket out folder. Both bucket folders are associated with this request (see p. 3.4).
- Document identifiers and metadata are send in request from USOS to eSignForStudy, which gets them one by one to be signed and sends back when the task is finished. There are two formats for parameters which gives two different protocols of that category.

5 Summary

HEIs in Poland use digital signatures for signing documents. However this process, due to its complexity and diversity of solutions, usually puts an extra burden on end users. eSignForStudy, which is well integrated with systems supporting student management and which allows the avoidance of hardware components at the user end, is a strongly appreciated benefit. Well designed and configured signature services which can be crafted to the requirements of the institution will not only ensure convenience but also efficiency.

The eSignForStudy solution is aligned with the strategies and activities at Polish national level, by supporting digitalization of the Higher Education Area, and also on a European level, by supporting cross-border exchange and validation of signed documents, thus increasing inter-operability and mutual recognition of electronic signatures across the EU.

When the solution is fully operational we will carry experiments to evaluate, from the user and institutional perspective, cost and effectiveness of various digital signing solutions, and prepare recommendations for HEIs – future users of the new system.

6 Acknowledgments

eSignForStudy project is co-funded by the CEF-TC-2020-1:eIdentification (eID) & eSignature programme under the grant 2020-EU-IA-0056. University of Warsaw is also co-financed by the program of the Minister of Science and Higher Education entitled "PMW" in the years 2021-2022; contract No. 5183/CEF/2021/2.

References

All links have been retrieved in March 2022.

[1] Digital Signature Service Documentation (2022), https://ec.europa.eu/cefdigital/DSS/webapp-demo/doc/dss-documentation.pdf.

[2] Erasmus Without Paper project website (2022), http://www.erasmuswithoutpaper.eu.

[3] eSignForStudy API specification (2022), (use https://validator.swagger.io to explore) https://dev.opteam.pl/ui/public/esignforstudy/openapi.json.

[4] eSignForStudy project website (2022), http://esignforstudy.eu.

[5] European Digital Student Service Infrastructure project website (2022), https://edssi.eu.

[6] Mincer-Daszkiewicz J. (2018). *Mobility scenarios supported by the Erasmus Without Paper Network* (full paper), EUNIS 2018, The 24th International Conference of European University Information Systems, 5-8 June 2018, Paris, France.

[7] Mincer-Daszkiewicz J. (2019). *Erasmus Without Paper Network — from development to production* (full paper in European Journal of Higher Education IT 2019-1, ISSN 2519-1764), EUNIS 2019, The 25th International Conference of European University Information Systems, 5-7 June 2019, Trondheim, Norway.

[8] MinIO website (2022), https://docs.min.io/docs.

[9] MUCI (Inter-University Centre for Informatization) website (2022), https://muci.edu.pl.

[10] OpenStack Swift wiki (2022), https://wiki.openstack.org/wiki/Swift.

[11] USOS API (2022), https://apps.usos.edu.pl/developers/api/.

[12] USOS (University Study-Oriented System) website (2022), https://www.usos.edu.pl/about-usos.

Author biographies

Janina Mincer-Daszkiewicz graduated in computer science in the University of Warsaw, Poland, and obtained a Ph.D. degree in math from the same university. She is an associate professor in Computer Science at the Faculty of Mathematics, Informatics and Mechanics at the University of Warsaw specializing in operating systems, distributed systems, performance evaluation and software engineering. Since 1999, she leads a project for the development of a student management information system USOS, which is used in 80 Polish Higher Education Institutions. Janina takes an active part in many nation-wide projects in Poland. She has been involved in Egracons, EMREX, Erasmus Without Paper, European Digital Student Service Infrastructure European projects.

Tadeus Gąsior graduated in math and computer science in the University of Rzeszów, Poland. Since 1997 he is connected with OPTeam SA. He took part in numerous projects, both as an architecture designer and multi technology programmer. Many of these projects were related to smart card personalization systems, Windows login with cards, bank e-purse, printing and copy management systems, Office 365 integration with various systems, PKI implementations, access control, data collection systems. Tadeusz is also Agile Scrum methodology evangelist, software auditor and consultant.