Politechnika Poznańska
Wydział Informatyki i Telekomunikacji
Instytut Informatyki
Politechnika Poznańska

Praca dyplomowa magisterska

ELEKTRONICZNA LEGITYMACJA STUDENCKA W WERSJI 2

Edyta Bosacka

Promotor
dr hab. inż. Marek Mika

Opiekun
mgr inż. Marek Gosławski

Poznań, 2020 r.

Karta pracy dyplomowej;

oryginał do wersji dla archiwum PP, w pozostałych kopiach ksero.

Streszczenie

W niniejszej pracy magisterskiej dokonano analizy teoretycznej zagadnień związanych z kartami elektronicznymi ze szczególnym uwzględnieniem Elektronicznej Legitymacji Studenckiej. Skrupulatnie opisane zostały zamieszczone w Rozporządzeniu Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. wymagania nałożone na nową wersję tego dokumentu. Ponadto omówiona została funkcjonalność mobilnej aplikacji „mLegitymacja" będącej alternatywą dla fizycznej formy legitymacji studenckiej. W celu wyboru właściwej technologii dla implementacji apletu ELS w wersji 2 porównane zostały różne wersje platformy Java Card. Celem niniejszej pracy było stworzenie wyżej wspomnianego apletu spełniającego wymogi Ministerstwa oraz zgodnego ze specyfikacją GlobalPlatform i normą ISO/IEC 7816. Wprowadzone rozwiązanie zostało pomyślnie przetestowane na fizycznych egzemplarzach kart elektronicznych.

Słowa kluczowe: Karty elektroniczne, ELS, Elektroniczna Legitymacja Studencka, mLegitymacja, Java Card, GlobalPlatform.

Abstract

In this master's thesis the theoretical analysis of issues related to electronic cards was performed with a particular emphasis on Electronic Student ID Card (ESC). The requirements imposed on the new version of this document included in Regulation of the Minister of Science and Higher Education of April 16 2019, were precisely described. Furthermore, the functionality of the mobile application „mLegitymacja" which is an alternative to the physical form of the student ID card, was described. In order to choose the relevant technology for the implementation of the ESC applet version 2, different version of the Java Card platform were compared. The goal of this master's thesis was an implementation of the above-mentioned applet that meets the requirements of the Ministry and complies with both the GlobalPlatform specification and ISO/IEC 7816 standard. The implemented solution was successfully tested on physical samples of smart cards.

Keywords: Smart cards, Electronic Student ID Card, mLegitymacja, Java Card, GlobalPlatform.

Spis Treści

1. Wstęp

Karty elektroniczne (ang. *smart cards*) stanowią jedno z większych osiągnięć w świecie technologii informatycznych. Od lat używane są jako specjalizowane przenośne komputery o kieszonkowych rozmiarach, służące do przechowywania i przetwarzania prywatnych informacji użytkowników, a także do komunikacji z innymi systemami komputerowymi za pośrednictwem połączeń przewodowych lub bezprzewodowych. Ze względu na gwarantowany poziom bezpieczeństwa, wygodę i niezawodność, karty te znalazły zastosowanie w wielu dziedzinach życia (m.in. w finansach, telekomunikacji i opiece zdrowotnej).

Spośród licznych dostępnych funkcjonalności kart elektronicznych, jedną z najczęściej wykorzystywanych, jest możliwość używania ich jako dokumentu poświadczającego tożsamość użytkownika. Korzyści płynące z takiego zastosowania kart zostały dostrzeżone przez Ministra Edukacji Narodowej i Sportu i dnia 18 lipca 2005 r. wydał on rozporządzenie [17] pozwalające na wprowadzenie na uczelniach wyższych legitymacji studenckiej w formie „elektronicznej karty procesorowej". Zdefiniowana w ten sposób Elektroniczna Legitymacja Studencka (ELS) mogła służyć nie tylko jako dokument tożsamości, ale również pełnić inne role. Z upływem czasu zakres funkcji pełnionych przez ELS rozszerzył się i dziś może być ona stosowana również jako karta dostępu do laboratoriów, urządzeń technicznych, karta biblioteczna, płatnicza, lojalnościowa czy elektroniczna portmonetka.

Ciągły rozwój technologii spowodował konieczność wprowadzenia zmian w dotychczasowej wersji Elektronicznej Legitymacji Studenckiej. Z tego powodu Minister Nauki i Szkolnictwa Wyższego wydał 16 kwietnia 2019 r. rozporządzenie [18] opisujące nową strukturę ELS oraz dopuszczające wykorzystanie tego dokumentu w formie aplikacji mobilnej, czyli tzw. mLegitymacji. Zgodnie z rozporządzeniem struktura ELS 2 powinna pozwalać na przechowywanie pliku ze zdjęciem studenta, a także wprowadzać nieznaczne modyfikacje w już istniejących plikach. Ponieważ Politechnika Poznańska zamierza w niedalekiej przyszłości wdrożyć ELS w wersji 2, niezbędne było dokonanie odpowiednich zmian w obecnie używanej implementacji Elektronicznej Legitymacji Studenckiej.

1.1 Cel i zakres pracy

Celem niniejszej pracy było zaimplementowanie apletu Elektronicznej Legitymacji Studenckiej w wersji 2. Stworzona aplikacja miała pozwalać na przechowywanie w swojej strukturze pliku z fotografią studenta oraz umożliwiać wykonywanie na nim podstawowych operacji zapisu i odczytu. Aplet ten powinien być możliwy do zastosowania nie tylko jako Elektroniczna Legitymacja Studencka, ale również jako Elektroniczna Legitymacja Doktorancka (ELD) oraz Elektroniczna Legitymacja Nauczyciela Akademickiego (ELNA). Wybór wariantu legitymacji oraz wersji struktury odbywać się będzie na etapie instalacji aplikacji na karcie procesorowej. W ramach pracy konieczne było również opracowanie zestawu testów funkcjonalnych, wykorzystujących polecenia do komunikacji z kartą - APDU, w celu sprawdzenia poprawności rozwiązania.

Praca składa się z 6 rozdziałów. W rozdziale 2 przedstawiono budowę oraz podział kart elektronicznych, a także opisano związane z nimi normy ISO. Rozdział 3 opisuje starą i nową wersję Elektronicznej Legitymacji Studenckiej oraz omawia „mLegitymację” – aplikację mobilną będącą alternatywą dla tego dokumentu. Dodatkowo rozdział ten porusza kwestie bezpieczeństwa związane z wprowadzeniem ELS w wersji 2. W rozdziale 4 znajduje się opis użytych w projekcie technologii: Java Card oraz GlobalPlatform. W następnym rozdziale przedstawione zostały wymagania związane z projektem, a także jego implementacja i opis przeprowadzonych testów. Rozdział 6 jest rozdziałem podsumowującym, zawierającym wnioski autora pracy.

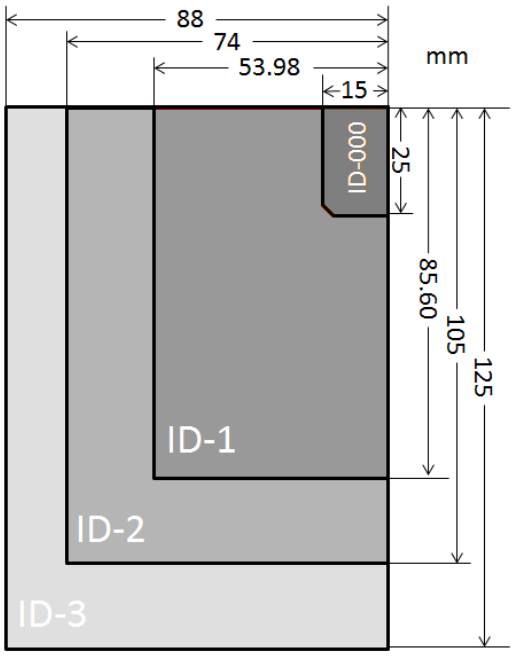
2. Karty elektroniczne

W rozdziale tym omówiono architekturę kart elektronicznych oraz ich podział ze względu na sposób przesyłania danych i rodzaj wbudowanego układu cyfrowego. Ponadto szczegółowo opisano wspólną dla wszystkich kart elektronicznych międzynarodową normę ISO 7816.

2.1 Architektura

Karty elektroniczne są przenośnymi urządzeniami pamięci masowej używanymi w różnych zastosowaniach wymagających kontrolowanego dostępu do informacji wrażliwych [1]. Karty te zwykle wykonane są z plastiku, zazwyczaj PCW, ale może być to również ABS lub poliwęglan [2]. Zgodnie z normą ISO/IEC 7810 rozróżnia się 4 standardowe rozmiary kart elektronicznych [3] (rysunek 2.1):

- **ID-1** 85.6 x 53.98 mm – format ten jest powszechnie stosowany w kartach płatniczych, prawach jazdy, a w niektórych państwach również w dowodach osobistych.
- **ID-2** 105 x 74 mm – format ten używany jest w wizach, a także np. w dowodach osobistych mieszkańców Francji i Rumunii.
- **ID-3** 125 x 88 mm – karty w tym rozmiarze stosowane są jako paszporty i wizy.
- **ID-000** 25 x 15 mm – format ten stosowany jest w kartach SIM.

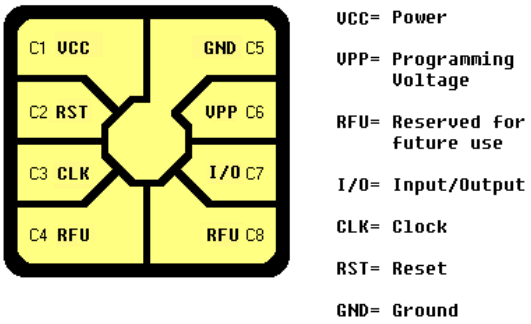


Rysunek 2.1 Wymiary kart elektronicznych wg ISO/IEC 7810.

Architektura kart elektronicznych zasadniczo składa się z trzech elementów [1]. Są to:

- Układ I/O
- Procesor (CPU)
- Pamięć

Karty elektroniczne muszą posiadać określone elementy do wykonywania operacji wejścia/wyjścia (I/O). Karty te zawierają wewnętrzny układ logiczny, który współpracuje z mikroprocesorem kontrolującym czas i przepływ danych przesyłanych do i z pamięci karty. Mają one fizyczną strukturę, która umożliwia im łączenie się z czytnikiem kart elektronicznych podłączonym do komputera-hosta w celu wymiany danych. Rodzaje fizycznych interfejsów zawartych w karcie elektronicznej zależą od jej rodzaju tzn. czy jest to karta stykowa czy bezstykowa. Interfejsem karty stykowej jest pole kontaktowe (rysunek 2.2), pod którym znajduje się moduł układu scalonego, natomiast interfejs karty bezstykowej zawiera, oprócz modułu układu scalonego antenę do komunikacji za pomocą fal radiowych.



Rysunek 2.2 Moduł układu scalonego na karcie elektronicznej [1].

Elementem, który odróżnia karty elektroniczne od zwykłych kart przeznaczonych do przechowywania danych jest procesor. Jego funkcję tradycyjnie pełni układ 8-bitowy, ale coraz częściej wykorzystywane są mocniejsze układy 16 i 32-bitowe [4]. Procesor wraz z systemem operacyjnym umożliwia karcie elektronicznej decydowanie o miejscu przechowywania danych w pamięci i o tym w jakich okolicznościach informacje powinny być przesyłane przez interfejs wejścia/wyjścia. Głównymi elementami, z których składa się procesor są:

- jednostka arytmetyczno-logiczna (ALU)
- jednostka sterująca
- magistrala

- zestaw rejestrów

Procesory kart elektronicznych wykonują instrukcje maszynowe z prędkością około 1 MIPS (milion instrukcji na sekundę). Często w celu poprawy szybkości obliczeń szyfrowania do układu dołączany jest koprocesor.

Pamięci wykorzystywane w mikrokontrolerach kart elektronicznych są wytwarzane z materiałów półprzewodnikowych. Składają się one z matryc komórek utworzonych przez tranzystory w celu przechowywania informacji. W kartach elektronicznych stosowane są trzy typy pamięci półprzewodnikowej:

- pamięć ROM
- pamięć RAM
- pamięć EEPROM

Pamięć ROM (pamięć tylko do odczytu) służy do przechowywania stałego oprogramowania karty [5]. Zawiera ona zarówno procedury systemu operacyjnego oraz trwałe dane. Proces zapisywania obrazu binarnego (reprezentującego programy i dane) nazywany jest *maskowaniem*. Zapis danych w tej pamięci odbywa się jedynie w trakcie produkcji karty, później nie istnieje już możliwość ich modyfikacji przez użytkownika. Przechowywanie danych w pamięci ROM nie wymaga zapewnienia ciągłego zasilania.

Tymczasową przestrzenią roboczą do przechowywania i modyfikowania danych w karcie elektronicznej jest pamięć RAM (pamięć o dostępie swobodnym). Pamięć ta jest pamięcią nietrwałą. Oznacza to, że informacje w niej zawarte są tracone wraz z odłączeniem zasilania. RAM pozwala na odczyt dowolnej komórki w pamięci, który można uzyskać w czasie od dziesiątek do setek nanosekund.

Pamięć EEPROM (elektrycznie kasowalno-programowalna pamięć tylko do odczytu), podobnie jak pamięć ROM, zachowuje dane nawet gdy zasilanie pamięci jest wyłączone. Różnica polega na tym, że zawartość tego rodzaju pamięci może być faktycznie modyfikowana podczas normalnego użytkowania karty. Pamięć ta jest zatem używana do przechowywania danych w sposób odpowiadający dyskowi twardemu w komputerze. W pamięci EEPROM mogą również być zapisywane aplikacje użytkownika. Do ważnych parametrów elektrycznych tej pamięci należą: liczba cykli zapisu w okresie żywotności karty, okres przechowywania danych oraz czas dostępu. EEPROM w większości kart elektronicznych może niezawodnie akceptować co najmniej 100 000 cykli zapisu oraz przechowywać dane przez okres 10 lat.

2.2 Podział kart elektronicznych

Karty elektroniczne można podzielić ze względu na sposób przesyłania danych na karty bezstykowe i stykowe, a także ze względu na rodzaj i możliwości wbudowanego w kartę układu – na karty pamięciowe i mikroprocesorowe [6].

Karty bezstykowe (tzw. karty zbliżeniowe) są to karty elektroniczne, w których transmisja danych między kartą a czytnikiem odbywa się bezkontaktowo – bez umieszczania karty w czytniku (rysunek 2.3b) [2]. W kartach tego typu wbudowany chip połączony jest z małą anteną, która służy do komunikacji za pomocą fal radiowych z anteną umieszczoną w czytniku. Dzięki temu karty te są wygodniejsze w użyciu, przyczyniają się do mniejszego zużycia elementów mechanicznych czytnika, a także są chronione przed fizycznymi uszkodzeniami spowodowanymi m.in. przyłożeniem zbyt wysokiego napięcia.

Drugim rodzajem kart są karty stykowe (rysunek 2.3a). W celu użycia takiej karty konieczne jest umieszczenie jej w czytniku w taki sposób, aby zapewnić bezpośrednie połączenie między stykami złącza czytnika a przewodzącą płytką stykową znajdującą się na powierzchni karty [8]. Zatem przekazywanie poleceń, danych oraz stanu karty odbywa się poprzez fizyczne pola kontaktowe.

Rysunek 2.3 Typy kart ze względu na sposób przesyłania danych: a) karta stykowa wraz z czytnikiem, b) karta bezstykowa wraz z czytnikiem [1].

Istnieją również karty, które mogą transmitować dane zarówno w sposób kontaktowy, jak i bezkontaktowy – są to tzw. karty hybrydowe.

Kartami pamięciowymi są karty elektroniczne składające się z układów pamięci oraz układu sterującego (rysunek 2.4a) [9]. Wyposażone są one w pamięć ROM zawierającą dane identyfikacyjne użytkownika oraz w pamięć EEPROM, w której

przechowywane są dane zewnętrznych aplikacji [10]. Zatem na kartach tego typu możliwe jest jedynie przechowywanie danych, ich zapis oraz odczyt. Ponieważ karty pamięciowe nie zawierają procesora, nie pozwalają one na przetwarzanie danych. Istnieją trzy rodzaje kart pamięci: prosta karta przeznaczona wyłącznie do przechowywania danych, chroniona karta umożliwiająca ograniczenie dostępu do operacji odczytu i zapisu w pamięci (przeważnie za pomocą hasła lub klucza systemowego) oraz karta przedpłacona (*ang. stored-value card*) zawierająca jednostki pamięci, których można użyć tylko raz (np. karta telefoniczna).

W przeciwieństwie do kart pamięciowych karty procesorowe oprócz bloków pamięci zawierają również wbudowany w kartę układ procesorowy (rysunek 2.4b) [9]. Działaniem procesora, jak i innych komponentów procesorowej karty elektronicznej zarządza system operacyjny karty. W odróżnieniu od systemów operacyjnych komputerów osobistych, system na karcie elektronicznej składa się z zaledwie kilku tysięcy bajtów kodu. Do zadań, które może wykonywać taki system należy transmisja danych przez dwukierunkowy interfejs szeregowy, ładowanie, obsługa i zarządzanie aplikacjami, przetwarzanie i kontrola wykonania poleceń z terminala, chroniony dostęp do danych, zarządzanie pamięcią i plikami oraz zarządzanie i wykonywanie algorytmów kryptograficznych.

Rysunek 2.4 Typy kart ze względu na rodzaj wbudowanego układu: a) karta pamięciowa, b) karta procesorowa [9].

2.3 Normy ISO

Karty elektroniczne wprowadzone zostały na masową skalę w wielu państwach świata. Z tego powodu zaistniała potrzeba ich ścisłej standaryzacji [10]. Najważniejszym standardem opisującym wymagania dot. kart elektronicznych jest norma ISO/IEC 7816 opracowana wspólnie przez Międzynarodową Organizację Normalizacyjną (ISO) i Międzynarodową Komisję Elektrotechniczną (IEC) [11]. Standard ISO/IEC 7816 podzielony jest na czternaście części.

2.3.1 Norma ISO/IEC 7816 – części od 7816-1 do 7816-4

W części pierwszej (ISO/IEC 7816-1) opisane zostały właściwości fizyczne kart z uwzględnieniem ich odporności na promieniowanie rentgenowskie, promieniowanie UV, statyczne pole elektryczne, pole elektromagnetyczne oraz temperaturę otoczenia [12]. Ponadto ISO/IEC 7816-1 definiuje wytrzymałość mechaniczną kart w celu zagwarantowania ich niezawodności w przewidywanym okresie użytkowania. Część druga (ISO/IEC 7816-2) określa wymiary, położenie, przeznaczenie, a także charakterystykę elektryczną metalowych styków karty. W części trzeciej (ISO/IEC 7816-3) przedstawiona została specyfikacja sygnałów oraz protokołów transmisji, które powinny zostać użyte do komunikacji z kartą. Z kolei najważniejszą częścią normy dla potrzeb programowania kart elektronicznych jest ISO/IEC 7816-4 [13]. Opisuje ona:

- zawartość par polecenie-odpowiedź wymienianych pomiędzy terminalem a kartą,
- sposób wyszukiwania elementów i obiektów danych w pamięci karty,
- struktury i zawartość historycznych bajtów,
- struktury dla aplikacji i danych na karcie,
- metody dostępu do plików i danych przechowywanych na karcie,
- środki i mechanizmy identyfikacji i adresowania aplikacji zapisanych na karcie,
- metody bezpiecznego przesyłania poleceń,
- metody dostępu do algorytmów przetwarzanych przez kartę.

Pary polecenie-odpowiedź

Komunikacja czytnika z kartą elektroniczną, po nawiązaniu połączenia i ustaleniu jego parametrów, przebiega z wykorzystaniem poleceń zwanych poleceniami APDU (*ang. Application Protocol Data Unit*). Po odebraniu polecenia następuje przetwarzanie na karcie i wysłanie odpowiedzi zwrotnej. ISO 7816-4 definiuje strukturę polecenia

APDU, której elementy przedstawione zostały w tabeli 2.1, gdzie N_c jest liczbą bajtów w polu danych polecenia, N_e jest maksymalną liczbą bajtów oczekiwaną w polu danych odpowiedzi, a N_r oznacza liczbę bajtów w polu danych odpowiedzi.

Pole	Opis	Liczba bajtów	Kierunek
Nagłówek polecenia	Bajt klasy oznaczony jako CLA	1	Do karty
	Bajt instrukcji oznaczony jako INS	1	
	Bajty parametrów oznaczone jako P1-P2	2	
Pole L_c	Brak przy $N_c = 0$, obecne dla $N_c > 0$	0, 1 lub 3	
Pole danych polecenia	Brak przy $N_c = 0$, występuje jako ciąg bajtów N_c, jeśli $N_c > 0$	N_c	
Pole L_e	Brak przy $N_e = 0$, obecne dla $N_e > 0$	0, 1, 2 lub 3	
Pole danych odpowiedzi	Brak przy $N_r = 0$, występuje jako ciąg bajtów N_r, jeśli $N_r > 0$	N_r (maks. N_e)	Z karty
Pole statusu	Bajty statusu oznaczone jako SW1-SW2	2	

Tabela 2.1 Elementy struktury poleceń APDU.

Bajt **CLA** wskazuje klasę polecenia. Służy on do określenia, w jakim stopniu polecenie i odpowiedź są zgodne z ISO/IEC 7816-4. W stosownych przypadkach CLA określa również format bezpiecznej wymiany komunikatów oraz numer kanału logicznego.

Bajt **INS** jest bajtem instrukcji, który wskazuje polecenie do przetworzenia. Zgodnie z ISO/IEC 7816-3 wartości bajta „6X” i „9X” są nieprawidłowe.

Bajty parametrów **P1-P2** określają elementy sterujące i opcje przetwarzania polecenia. Mogą one przyjmować dowolną wartość.

N_c oznacza liczbę bajtów w polu danych polecenia. Pole L_c służy do kodowania wartości N_c:

- Jeśli $N_c = 0$, wówczas nie ma pola L_c,
- Krótka wersja pola L_c składa się z jednego bajta różnego od „00” (N_c o wartości od 1 do 255).

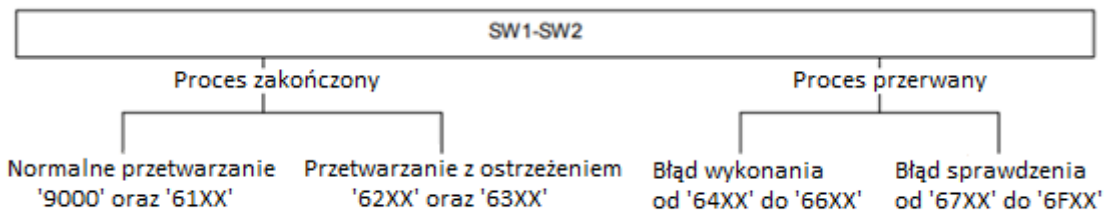
- Rozszerzona wersja pola L_c składa się z trzech bajtów: jednego bajta ustawionego na „00”, po którym następują dwa bajty od „0001” do „FFFF” (N_c o wartości od 1 do 65 535).

N_e oznacza maksymalną liczbę bajtów oczekiwaną w polu danych odpowiedzi, niezależnie od struktury danych zawartych w tym polu. Pole L_e służy do kodowania wartości N_e:

- Jeśli $N_e = 0$, wówczas nie ma pola L_e,
- Krótka wersja pola L_e składa się z jednego bajta o dowolnej wartości: „00” dla N_e o wartości 256, od „01” do „FF” dla N_e o wartości od 1 do 255.
- Rozszerzona wersja pola L_e składa się albo z trzech bajtów (jeden bajt ustawiony na „00”, po którym następują dwa bajty z dowolną wartością), jeśli nie ma pola L_c albo z dwóch bajtów (z dowolną wartością), jeśli obecna jest rozszerzona wersja pola L_c:
    - jeśli te dwa bajty ustawione są na „0000”, wówczas N_e ma wartość 65 536,
    - dla wartości bajtów od „0001” do „FFFF” N_e przyjmuje wartość od 1 do 65 535.

N_r oznacza liczbę bajtów w polu danych odpowiedzi. N_r powinno być mniejsze lub równe N_e. Dlatego w każdej parze polecenie-odpowiedź brak pola L_e jest standardowym sposobem otrzymywania pola danych bez odpowiedzi.

Bajty **SW1-SW2** wskazują status przetwarzania. Zgodnie ze specyfikacją ISO/IEC 7816-3 wszystkie wartości różne od „6XXX” oraz „9XXX” są nieprawidłowe. Niepoprawne są również wartości „60XX”. Na rysunku 2.5 przedstawiony został schemat strukturalny dla wartości „9000” i od „61XX” do „6FXX” dla bajtów SW1-SW2.



Rysunek 2.5 Schemat strukturalny wartości bajtów SW1-SW2 [13].

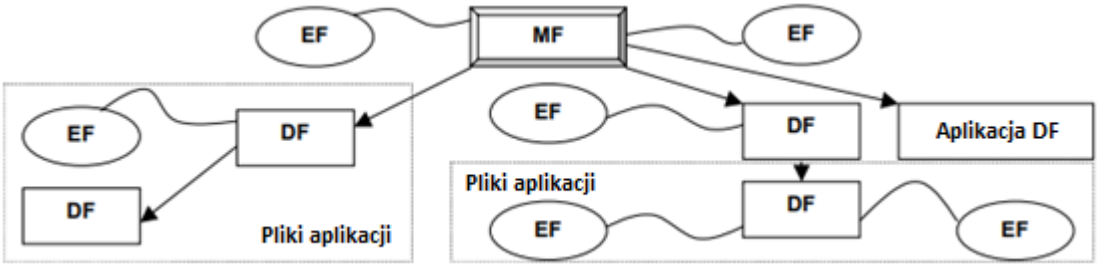
Struktury dla danych i aplikacji

ISO/IEC 7816-4 definiuje struktury danych i aplikacji znajdujących się na karcie. Do struktur wspieranych przez tę normę należą:

- **Plik dedykowany (DF) –** zawiera aplikacje i obiekty danych oraz grupuje pliki. Plik DF może być rodzicem dla innych struktur, których typy powinny należeć do następującego zbioru {DF, EF, DO}.
- **Plik elementarny (EF) –** służy do przechowywania danych. Plik EF może być rodzicem dla innych struktur o następujących typach: {DO, Rekord, DataString}. W normie określono dwie kategorie plików EF:
    - pliki wewnętrzne EF przechowujące dane interpretowane przez kartę, tj. dane wykorzystywane przez kartę do celów zarządzania i kontroli.
    - pliki robocze EF przechowujące dane, które nie są interpretowane przez kartę, tj. dane używane wyłącznie przez aplikacje zewnętrzne.
- **Rekord –** służy do przechowywania danych. Rekord może być rodzicem innych struktur, które są obiektami danych (DO).
- **DataString –** służy do przechowywania danych. DataString jest sekwencją bajtów w pliku transparentnym EF. Może być on rodzicem innych struktur, które są obiektami danych (DO).
- **Data Object (DO)** – służy do przechowywania danych. Data Object może być rodzicem innych obiektów danych.

Część czwarta standardu ISO/IEC 7816 definiuje dwie organizacje logiczne plików na karcie elektronicznej:

- Hierarchia plików DF wraz z odpowiadającą jej architekturą bezpieczeństwa (rysunek 2.6). W takiej organizacji plików DF w katalogu głównym nazywany jest plikiem MF (*ang. Master File*). Każdy plik DF może zawierać aplikację oraz własną hierarchię plików DF.
- Organizacja równoległa plików DF z aplikacjami bez widocznej hierarchii pomiędzy plikami DF (bez pliku MF) (rysunek 2.7). Taka organizacja wspiera niezależne aplikacje na karcie, gdyż każdy plik DF może mieć własną hierarchię plików oraz architekturę bezpieczeństwa.



Rysunek 2.6 Organizacja hierarchiczna plików DF [13].

Rysunek 2.7 Organizacja równoległa niezależnych aplikacji DF [13].

Obiekty danych oraz sposób ich wyszukiwania na karcie

Istnieją dwa rodzaje obiektów danych: **SIMPLE-TLV** oraz **BER-TLV**. Każdy obiekt SIMPLE-TLV powinien składać się z dwóch lub trzech kolejnych pól: obowiązkowego pola znacznika, obowiązkowego pola długości i warunkowego pola wartości. Pole znacznika składa się z jednego bajta kodującego numer znacznika od 1 do 254. Wartości „00” i „FF” są nieprawidłowe dla pól znaczników. Pole długości składa się z jednego lub trzech kolejnych bajtów:

- Jeśli pierwszy bajt jest różny od „FF”, wówczas pole długości składa się z pojedynczego bajta o wartości od 0 do 254 (oznaczonej jako N).
- Jeśli pierwszy bajt jest ustawiony na „FF”, wówczas pole długości jest kontynuowane na dwóch kolejnych bajtach o dowolnej wartości od 0 do 65 535 (oznaczonej jako N).

Jeśli N wynosi 0, wtedy nie ma pola wartości tzn. obiekt danych jest pusty. W przeciwnym wypadku (N > 0) pole wartości składa się z N kolejnych bajtów.

Analogicznie każdy obiekt BER-TLV musi składać się z dwóch lub trzech kolejnych pól: obowiązkowego pola znacznika, obowiązkowego pola długości oraz warunkowego pola wartości. Pole znacznika składa się z co najmniej jednego bajta. Wskazuje ono klasę, typ kodowania oraz koduje numer znacznika. Pole długości składa się z od jednego do pięciu kolejnych bajtów kodujących długość obiektu (oznaczoną jako N). Jeśli N wynosi 0, wówczas nie ma pola wartości tzn. obiekt danych jest pusty i oznaczony jako {T-‘00’}. Jeśli N > 0 pole wartości składa się z N kolejnych bajtów,

a obiekt danych jest oznaczony jako {T-L-V}, gdzie T jest numerem znacznika, L jest długością pola wartości, a V oznacza wartość.

2.3.2 Norma ISO/IEC – części od 7816-5 do 7816-9

W ISO/IEC 7816-5 wyznaczone zostały standardy dla identyfikatorów aplikacji (*ang. Application Identifier - AID*). Każdy identyfikator AID musi składać się z dwóch części. Pierwszą z nich stanowi unikalny zarejestrowany identyfikator dostawcy aplikacji (RID) o długości pięciu bajtów. Drugą częścią jest pole o zmiennej długości do 11 bajtów, którego dostawcy aplikacji mogą używać do identyfikacji określonych aplikacji. W kolejnej części - ISO/IEC 7816-6 opisano elementy danych używane do wymiany międzybranżowej w oparciu zarówno o karty stykowe, jak i bezstykowe. Dla każdego elementu danych zdefiniowano następujące właściwości: identyfikator, nazwa, opis, format i kodowanie, a także określono metody wyszukiwania elementów danych na karcie. Część siódma - ISO/IEC 7816-7 zawiera opis poleceń międzybranżowych dla języka SCQL (*ang. Structured Card Query Language*). W następnej części ISO/IEC 7816-8 zdefiniowano wewnętrzne polecenia karty dla operacji bezpieczeństwa. Polecenia te są komplementarne i opierają się na komendach wymienionych w normie ISO/IEC 7816-4. W części dziewiątej - ISO/IEC 7816-9 określono polecenia wykorzystywane przy zarządzaniu kartą, plikami oraz innymi strukturami, tj. obiekty danych i bezpieczeństwa [14]. Polecenia te obejmują cały cykl życia karty, dlatego niektóre z nich są używane przed wydaniem karty lub po przekroczeniu jej terminu ważności. Do podstawowych stanów cyklu życia obiektu na karcie elektronicznej należą:

- Stan utworzenia – obiekt jest nowo tworzony (np. za pomocą poleceń CREATE lub CREATE FILE) lub dołączany do istniejącego obiektu (np. za pomocą poleceń UPDATE DATA lub PUT DATA).
- Stan inicjalizacji – nowo utworzony obiekt lub istniejący obiekt w stanie utworzenia może zostać zainicjalizowany. Obiekt ten nie jest aktywny, ale może zostać wybrany i wypełniony danymi.
- Stan operacyjny – składa się z dwóch podstanów: stanu aktywnego i nieaktywnego. W stanie aktywnym do obiektu i jego zawartości można uzyskać dostęp zgodnie z jego ustawieniami bezpieczeństwa. Natomiast w stanie nieaktywnym obiekt ma ograniczoną funkcjonalność, a dostęp do jego zawartości zależy od określonej aplikacji.

- Stan zakończenia – w tym stanie jedynym dozwolonym poleceniem jest usunięcie obiektu, o ile aplikacja nie określiła inaczej.
- Stan zakończenia używania karty – po udanym wykonaniu polecenia TERMINATE CARD USAGE karta powinna odrzucać polecenie SELECT.

2.3.3 Norma ISO/IEC – części od 7816-10 do 7816-15

Kolejna część ISO/IEC 7816-10 określa moc, strukturę sygnałów oraz strukturę odpowiedzi na reset (*ang. Answer To Reset – ATR*) przesyłaną pomiędzy układem scalonym karty a urządzeniem z interfejsem, tj. terminalem. Część ta opisuje również szybkość sygnałów, warunki działania oraz komunikację z układem scalonym karty. W ISO/IEC 7816-11 zawarto z kolei polecenia związane z bezpieczeństwem, które mają na celu weryfikację użytkownika na podstawie jego cech biometrycznych. Norma ta definiuje struktury danych oraz metody dostępu do nich w celu wykorzystania karty jako nośnika danych biometrycznych jej posiadacza. W części dwunastej – ISO/IEC 7816 -12 określono warunki działania układu scalonego karty z interfejsem USB (*ang. USB-ICC*). Standard ten w szczególności opisuje: charakterystykę elektryczną interfejsu USB-ICC, deskryptory USB i USB-ICC, wymianę danych przy użyciu transferów masowych lub kontrolnych, a także statusy i warunki błędów. ISO/IEC 7816-13 zawiera polecenia pozwalające na zarządzanie wieloma aplikacjami na karcie. Natomiast ostatnia część normy – ISO/IEC 7816-15 dotyczy aplikacji zawierającej informacje na temat funkcjonalności kryptograficznej. Część ta definiuje wspólną składnię i format oraz mechanizmy udostępniania informacji kryptograficznych w stosownych przypadkach.

3. Elektroniczna Legitymacja Studencka

W rozdziale tym szczegółowo opisano dotychczas używaną wersję Elektronicznej Legitymacji Studenckiej, a także jej nową wersję będącą przedmiotem niniejszej pracy. Dodatkowo omówiono aplikację mobilna zwaną „mLegitymacją" stanowiącą nowoczesną alternatywę dla tradycyjnej formy tego dokumentu. Na koniec poruszono kwestie bezpieczeństwa związane z wprowadzeniem nowej struktury ELS.

3.1 Wersja 1 ELS

Elektroniczna Legitymacja Studencka w wersji 1 została określona po raz pierwszy w „Rozporządzeniu Ministra Edukacji Narodowej i Sportu z dnia 18 lipca 2005 r. w sprawie dokumentacji przebiegu studiów" [15]. Dokument ten opisuje wymiary oraz dokładny projekt graficzny takiej karty, a także jej funkcje. Poza pełnieniem roli legitymacji studenckiej, karta ta może być również używana jako karta komunikacji miejskiej, karta dostępu do pomieszczeń i zasobów uczelni lub jako karta biblioteczna. Mnogość możliwych zastosowań wymusza potrzebę zapewnienia wiarygodności ELS jako dokumentu poświadczającego aktualny status studenta. Zostało to osiągnięte dzięki wgraniu na kartę unikalnych danych studenta wraz z podpisem elektronicznym weryfikowanym przy użyciu ważnego certyfikatu kwalifikowanego.

Opis ELS w wersji 1 jest również częścią aktualnego „Rozporządzenia Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. zmieniającego rozporządzenie w sprawie studiów" [18]. Zgodnie z nim standardowe wymiary karty elektronicznej wynoszą 85.6 mm x 53.98 mm i odpowiadają one karcie identyfikacyjnej ID-1 zgodnie z normą ISO/IEC 7810. Na karcie, w procesie personalizacji graficznej, drukowane są następujące dane osobowe studenta: imię i nazwisko, numer albumu, PESEL, adres oraz zdjęcie, a także informacje o uczelni, która wydała kartę oraz dacie wydania. Projekt Elektronicznej Legitymacji Studenckiej w wersji 1 pokazany został na rysunku 3.1.

Rysunek 3.1 Wzór Elektronicznej Legitymacji Studenckiej w wersji 1 [16].

Na część elektroniczną ELS składa się karta procesorowa z interfejsem stykowym określonym w normach ISO/IEC 7816-2 oraz ISO/IEC 7816-3. ELS może posiadać także inne interfejsy, włączając w to interfejs bezstykowy.

Struktura danych znajdujących się na karcie jest zgodna z czwartą częścią normy ISO/IEC 7816 [17]. Rozporządzenie określa, że w pamięci ELS obowiązkowo musi być zawarty główny plik DF.SELS oraz jego dwa pliki potomne – EF.CERT i EF.ELS. Bezpośrednio w pliku DF.SELS zawarte są dane związane z legitymacją studencką. Jest to plik dedykowany o nazwie, która jest zarejestrowanym w Polskim Komitecie Normalizacyjnym identyfikatorem aplikacji. Własne rozszerzenie tego identyfikatora (PIX) dla ELS jest równe „01 01”. Wymagane jest, aby plik DF.SELS był dostępny za pomocą polecenia wyboru – SELECT FILE po każdym wykonanym resecie karty. W pliku potomnym EF.CERT o identyfikatorze 00 01 zapisany jest certyfikat kwalifikowany podpisu elektronicznego lub pieczęci elektronicznej. W polu „Właściciel certyfikatu” tego pliku zawarte są następujące atrybuty: nazwa powszechna (w tym atrybucie wpisane jest sformułowanie „osoba upoważniona do wystawiania legitymacji studenckiej”), nazwa organizacji, nazwa województwa, nazwa miejscowości oraz adres (atrybuty te dotyczą uczelni). Z kolei plik potomny EF.ELS o identyfikatorze 00 02 zawiera wiadomość podpisaną certyfikatem kwalifikowanym. Wiadomość ta zawiera kilka istotnych danych na temat studenta korzystającego z elektronicznej legitymacji studenckiej. Dane te zapisane są w strukturze SignedData i składają się na nie następujące atrybuty: wersja ELS, numer seryjny układu scalonego na karcie, nazwa uczelni, nazwisko studenta, imiona studenta, numer albumu, numer edycji legitymacji, numer pesel studenta oraz data ważności legitymacji. Na listingu 3.1 przedstawiono kod definiujący strukturę SELSInfo zawierającą wymienione atrybuty.

```
SELSInfo ::=SEQUENCE {
 wersja                  INTEGER{v1(1)}
 numerSeryjnyUkladu      PrintableString(SIZE (8..16)),
 nazwaUczelni            UTF8String (SIZE (1..128)},
 nazwiskoStudenta        SEQUENCE OF
                         UTF8String (SIZE (1..28)),
 imionaStudenta          SEQUENCE OF
                         UTF8String (SIZE (1..24)),
 numerAlbumu             PrintableString (SIZE (1..16)),
 numerEdycji             PrintableString (SIZE (1)),
 numerPesel              PrintableString (SIZE (11)),
 dataWaznosci            GeneralizedTime
 },
```

Listing 3.1 Struktura SELSInfo dla ELS w wersji 1.

3.2 Struktura ELS 2 w odniesieniu do ELS 1

Elektroniczna Legitymacja Studencka w wersji 2 została wprowadzona wraz z „Rozporządzeniem Ministra Nauki i Szkolnictwa Wyższego zmieniającym rozporządzenie w sprawie studiów z dnia 16 kwietnia 2019 r.” [18]. Nowa wersja ELS nie różni się pod względem wymiarów i wykonania od wersji 1. Jedyną widoczną różnicą w projekcie graficznym karty jest brak naniesionej informacji na awersie legitymacji o adresie zamieszkania studenta. Projekt Elektronicznej Legitymacji Studenckiej w wersji 2 pokazany został na rysunku 3.2 [18].

Rysunek 3.2 Wzór Elektronicznej Legitymacji Studenckiej w wersji 2 [18].

Struktura danych zapisanych na karcie procesorowej ELS 2 podobnie jak w wersji 1 jest zgodna z normą ISO/IEC 7816-4. W pamięci, analogicznie do wersji poprzedniej, znajduje się obowiązkowo plik DF.SELS oraz jego pliki potomne EF.CERT i EF.ELS. Dodatkowo jednak na karcie procesorowej może znajdować się trzeci plik potomny – EF.PHOTO. W pliku tym znajduje się cyfrowy zapis fotografii studenta w formacie JPEG, która fizycznie umieszczona jest na awersie legitymacji studenckiej. Dane zawarte w plikach DF.SELS i EF.CERT są identyczne w obu wersjach ELS. Struktura danych w pliku EF.ELS została jednak wzbogacona o kilka atrybutów m.in. o pole związane ze zdjęciem studenta. Atrybuty te zapisane są w strukturze SignedData, której postać przedstawiona jest na listingu 3.2:

```
SELSInfo ::=SEQUENCE
 wersja                  INTEGER v2(2)
 numerSeryjnyUkladu      PrintableString(SIZE (8..16)),
 nazwaUczelni            UTF8String (SIZE (1..128)},
 nazwiskoStudenta        SEQUENCE OF
                         UTF8String (SIZE (1..28)),
 imionaStudenta          SEQUENCE OF
                         UTF8String (SIZE (1..24)),
 numerAlbumu             PrintableString (SIZE (1..16)),
 numerEdycji             PrintableString (SIZE (1)),
 numerPesel              PrintableString (SIZE (11)),
 dataWaznosci            GeneralizedTime,
 dataWydania             GeneralizedTime,
 urlUnieważnienia        UTF8String (SIZE (1..128)),
 funkcjaSkrotu           OBJECT IDENTIFIER,
 skrotZdjecia            BIT STRING,
 efPhotoId               OCTET STRING
```

Listing 3.2 Struktura SELSInfo dla ELS w wersji 2.

Znaczenie poszczególnych pól jest następujące:

- `wersja` – w polu tym zapisany jest numer wersji struktury podpisywanych danych; obecność tego pola ułatwia rozpoznanie możliwych nowych wersji struktur danych w ELS;

- `numerSeryjnyUkladu` – w polu tym znajduje się zapisany w formacie heksadecymalnym unikalny numer seryjny układu nadany przez producenta; gdy w pamięci karty zapisywane są dane, aplikacja, która dokonuje zapisu, sprawdza zgodność zawartości tego pola z numerem seryjnym chipu odczytanym z ELS;
- `nazwaUczelni` – w polu tym zapisana jest formalna nazwa uczelni;
- `nazwiskoStudenta` i `imionaStudenta` – pola te wypełnione są zgodnie z danymi znajdującymi się w dowodzie osobistym lub paszporcie studenta;
- `numerAlbumu` – pole z unikalnym numerem nadanym studentowi przez uczelni;
- `numerEdycji` – w polu tym znajduje się literowe oznaczenie legitymacji studenckiej; pierwsza wydana legitymacja jest oznaczona literą A, a kolejne wydane duplikaty literami B, C, D itp.;
- `numerPesel` – pole z unikalnym numerem z Powszechnego Elektronicznego Systemu Ewidencji Ludności;
- `dataWaznosci` – w polu tym zapisana jest data upływu terminu ważności elektronicznej legitymacji studenckiej; wartość tego pola jest modyfikowana co semestr wraz z umieszczeniem hologramu w kolejno oznaczonych polach na rewersie legitymacji;
- `dataWydania` – w polu tym zapisana jest data wydania elektronicznej legitymacji studenckiej, data ta jest zgodna z datą wydania umieszczoną na awersie legitymacji;
- `urlUniewaznienia` – pole to zawiera adres URL, w którym przechowywana jest informacja o unieważnieniu legitymacji; w przypadku legitymacji unieważnionej obowiązkowo pod tym adresem musi być umieszczony ciąg znaków „UNIEWAŻNIONA”, dla legitymacji ważnej dane odczytane z tego adresu mogą być równe łańcuchowi znaków „WAŻNA”, lecz nie jest to wymagane;
- `funkcjaSkrotu` – w polu tym znajduje się identyfikator obiektu zawierającego funkcję skrótu; funkcja ta jest używana do obliczania wartości umieszczonej w polu skrotZdjecia, przykładowo: dla SHA-256 joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2) sha256(1);
- `skrotZdjecia` – pole to zawiera wartość skrótu zdjęcia umieszczonego w pliku EF.PHOTO, obliczoną z wykorzystaniem algorytmu zapisanego w polu funkcjaSkrotu;
- `efPhotoId` – w polu tym znajduje się składający się z dwóch bajtów identyfikator pliku potomnego EF.PHOTO.

3.3 Mechanizmy mLegitymacji

W rozdziale tym omówiono aplikację mobilną „mLegitymacja", opisano jej funkcjonalność oraz przedstawiono specjalny portal przeznaczony do jej obsługi.

3.3.1 Cyfrowa wersja legitymacji studenckiej - mLegitymacja

Aplikacja mobilna mLegitymacja jest wersją elektronicznej legitymacji studenckiej, która została wprowadzona w ramach „Rozporządzenia Ministra Nauki i Szkolnictwa Wyższego zmieniającego rozporządzenie w sprawie studiów z dnia 16 kwietnia 2019 r." [18][19]. Jest ona jednym z tzw. „mDokumentów", które są przechowywane na urządzeniu mobilnym jako część aplikacji mObywatel dostarczanej przez Ministra Cyfryzacji. Celem mLegitymacji jest zapewnienie studentom bezpiecznego, trudnego do podrobienia narzędzia, które będzie pozwalało na zweryfikowanie informacji umieszczonych na elektronicznej legitymacji studenckiej. Okazanie mLegitymacji pozwala studentowi na korzystanie z takich samych uprawnień jak w przypadku fizycznej legitymacji np. na korzystanie ze zniżek przy przejazdach komunikacją miejską, kolejową itp.

Rozporządzenie z dnia 16 kwietnia 2019 r. zawiera opis słowny informacji wymaganych w mLegitymacji studenckiej, a także wizualizację graficzną ważnej oraz nieważnej wersji tego dokumentu [18]. Do elementów wizualizacji ważnej mLegitymacji na urządzeniu mobilnym należą:

- czerwony nagłówek aplikacji z napisem „Legitymacja studencka",
- hologram przedstawiający godło Rzeczypospolitej Polskiej, w którym kolory tła zmieniają się wraz z kątem pochylenia urządzenia mobilnego,
- napis „RZECZPOSPOLITA POLSKA" znajdujący się obok hologramu,
- flaga państwowa przedstawiona jako animacja biało-czerwonej flagi powiewającej na wietrze,
- napis „Nr albumu: [n]", w którym [n] jest unikalnym numerem nadanym studentowi przez uczelnię,
- napis „Wydana: [d]", w którym [d] oznacza datę wydania legitymacji,
- obszar danych studenta na który składają się:
    a) kolorowa fotografia,
    b) imię lub imiona,
    c) nazwisko,
    d) etykieta określająca płeć: „Student" lub „Studentka"

e) napis „Data urodzenia: [d]”, w którym [d] oznacza datę urodzenia studenta,

f) napis „PESEL: [p]”, w którym [p] jest numerem PESEL studenta,

g) napis „[n] lat” lub „[n] lata”, w którym [n] określa wiek studenta,

- obszar z informacją o ważności legitymacji studenckiej (rysunek 3.3 i 3.4)

a) symbol „✓” w przypadku ważnej mLegitymacji lub „✗” w przypadku nieważnej mLegitymacji,

b) napis „Ważna” lub „Nieważna”,

c) napis „Termin ważności: [d]”, w którym [d] oznacza datę ważności mLegitymacji zgodną z datą ważności ELS,

- obszar z danymi dotyczącymi uczelni zawierający:

a) nagłówek z napisem „Nazwa uczelni”,

b) wpis z nazwą uczelni,

- napis „Czas okazania: [d]”, w którym [d] oznacza datę, godzinę i minutę okazania mLegitymacji zgodną z ustawieniami czasu w urządzeniu mobilnym,

- napis „Poświadcza uprawnienia do 50% ulgi przy przejazdach środkami komunikacji miejskiej, a także uprawnienia do korzystania – do ukończenia 26 roku życia – z ulgowych przejazdów środkami publicznego transportu zbiorowego autobusowego i kolejowego na podstawie odrębnych przepisów”.

Rysunek 3.3 Wzór ważnej mLegitymacji [18].

Rysunek 3.4 Wzór nieważnej mLegitymacji [18].

3.3.2 Funkcje mLegitymacji

W ramach aplikacji mLegitymacja użytkownik uzyskuje dostęp do kilku funkcji [20]. Podstawową z nich jest możliwość okazania mLegitymacji na ekranie urządzenia mobilnego. Dzięki temu osoba uprawniona do kontroli legitymacji ma możliwość ocenienia prawdziwości okazanego dokumentu na podstawie wcześniej opisanych elementów wizualnych. Kolejną dostępną funkcją jest przekazywanie danych z usługi mLegitymacja do aplikacji mWeryfikator. W tym celu użytkownik musi wybrać funkcję „Przekaż" w aplikacji. Następnie na ekranie urządzenia wyświetlony zostanie kwadratowy kod graficzny (kod QR), który osoba weryfikująca powinna zeskanować w przeciągu 10 minut za pomocą aparatu fotograficznego swojego urządzenia mobilnego. Zeskanowanie kodu QR powoduje nawiązanie połączenia Bluetooth między urządzeniami w celu przekazania danych z mLegitymacji. Po pomyślnym wykonaniu tej operacji w aplikacji mWeryfikator wyświetlone zostaną następujące dane dotyczące studenta: numer legitymacji i data jej wydania, imię (imiona) i nazwisko studenta, wiek, termin ważności mLegitymacji, nazwa oraz adres uczelni, a także fotografia studenta w niskiej rozdzielczości. Osoba weryfikująca na podstawie przesłanych danych ma możliwość zweryfikowania ich poprawności, a dodatkowo może ona sprawdzić aktualność certyfikatu online. Wszystkie operacje weryfikacji danych są zapisywane w „Historii" aplikacji mLegitymacja. Funkcja ta pozwala na sprawdzenie poprzednich weryfikacji wraz z ich dokładną datą i czasem oraz identyfikatorem aplikacji mWeryfikator. Wszystkie dane w „Historii" są przechowywane przez okres 1 roku od daty ich zapisania w usłudze. Z poziomu aplikacji użytkownik ma również możliwość usunięcia wszystkich danych zawartych w mLegitymacji oraz unieważnienia certyfikatu. Po wykonaniu tej funkcji, aby aplikacja mogła być ponownie używana, wymagana jest jednak jej ponowna aktywacja.

Unieważnienia mLegitymacji może dokonać wyłącznie uczelnia wydająca elektroniczną legitymację studencką. Ma to miejsce w następujących okolicznościach:

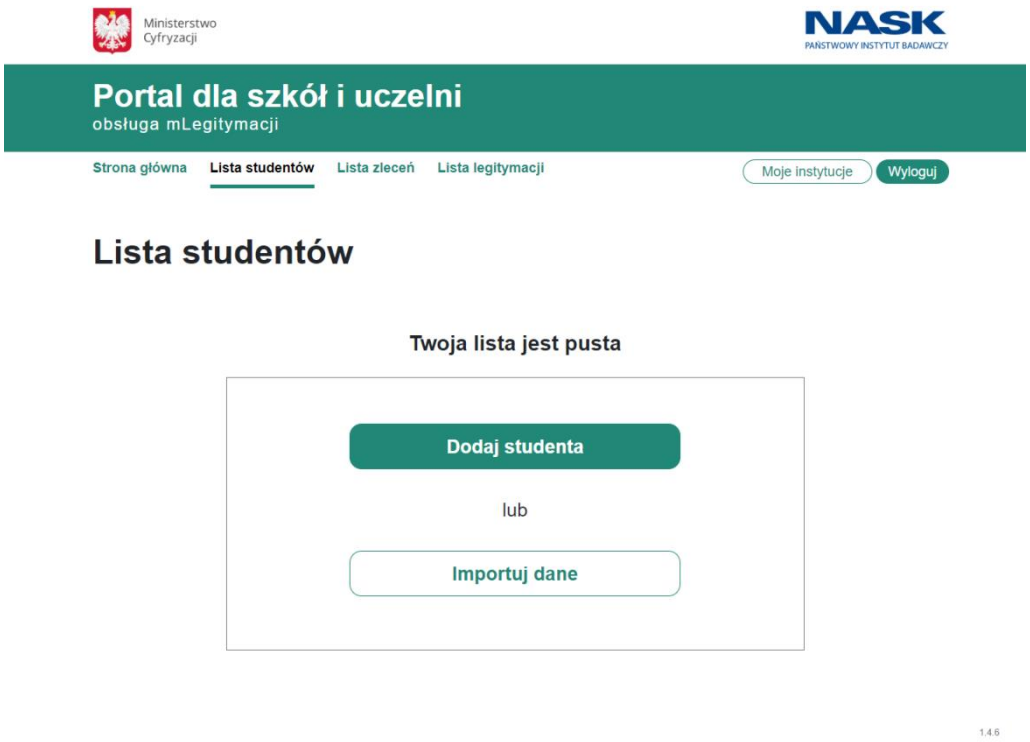
- w przypadku przekroczenia daty ważności wydanej legitymacji,
- na skutek uszkodzenia, nieprawidłowego działania lub straty urządzenia mobilnego z mLegitymacją,
- w przypadku zmiany uczelni przez studenta,
- na prośbę studenta.

3.3.3 Portal do obsługi mLegitymacji

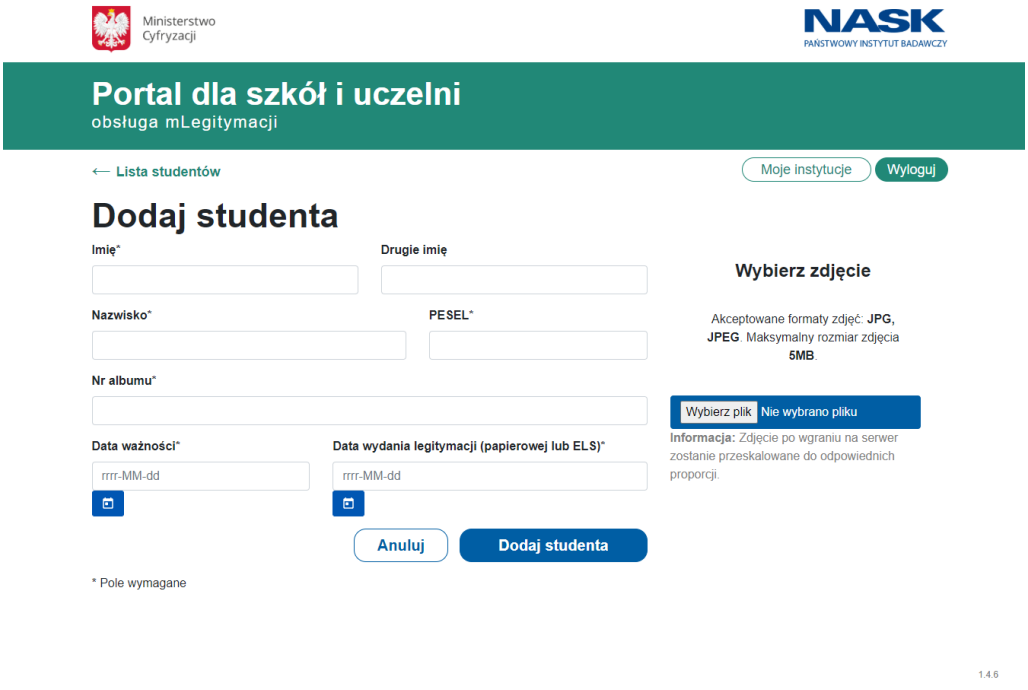
Aby umożliwić wydawanie mLegitymacji studenckich Ministerstwo Cyfryzacji stworzyło specjalny system – Portal dla szkół i uczelni (rysunek 3.5) [21]. Portal ten pozwala na wprowadzenie danych studentów i wygenerowanie dla nich mLegitymacji, która następnie może zostać aktywowana w aplikacji mObywatel. Osoba obsługująca system najczęściej realizuje następujące procedury: wydanie legitymacji, zastrzeżenie legitymacji oraz przedłużenie legitymacji.

Rysunek 3.5 Strona główna Portalu dla szkół i uczelni [21].

W celu wydania mLegitymacji użytkownik musi najpierw zalogować się do systemu przy użyciu Profilu Zaufanego. Jeśli go nie posiada, powinien skorzystać z funkcji „ Załóż Profil Zaufany”. Gdy użytkownik już się zaloguje, może on rozpocząć dodawanie lub importowanie osób do listy studentów. W tym celu musi on przejść do zakładki „Lista studentów”, a następnie wybrać jedną z opcji – „Dodaj studenta” lub „Importuj dane” (rysunek 3.6). W przypadku opcji importowania dane studentów należy najpierw przygotować w specjalnym arkuszu kalkulacyjnym. Następnie w arkuszu tym wywoływana jest funkcja generowania pliku leg.xml, który zawiera dane dotyczące studentów. Plik ten musi zostać wybrany po naciśnięciu przycisku „Importuj dane”. Jeśli użytkownik wybierze opcję „Dodaj studenta”, będzie on musiał ręcznie wprowadzić wszystkie wymagane dane (rysunek 3.7).



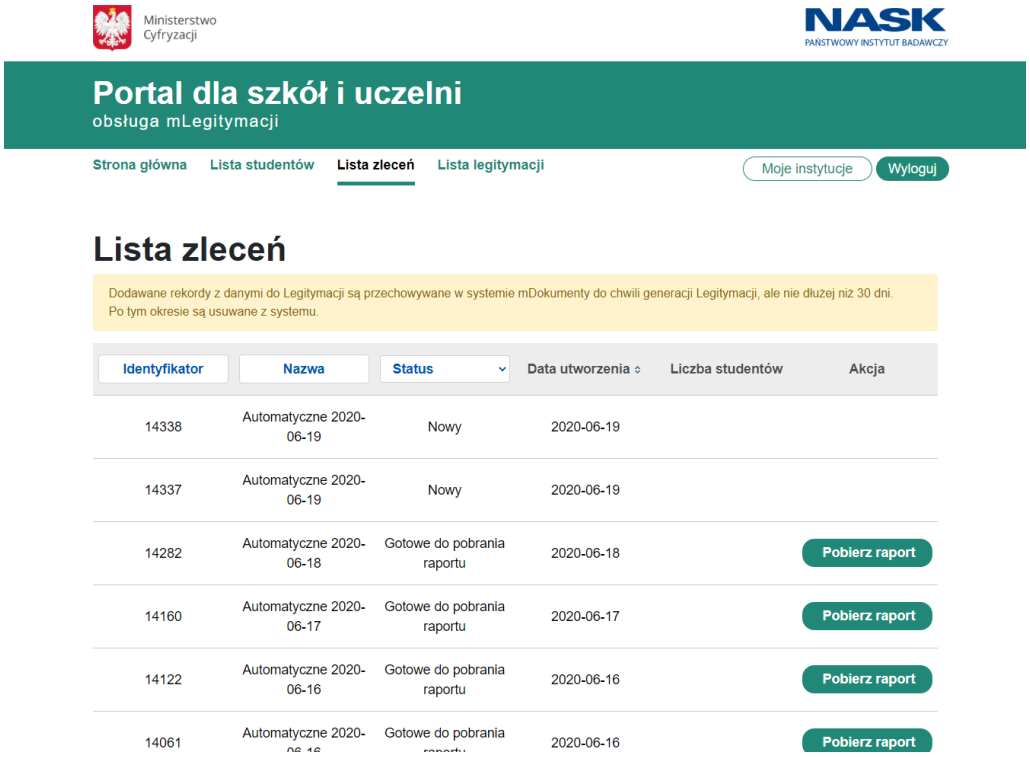
Rysunek 3.6 Zakładka „Lista studentów” [21].



Rysunek 3.7 Dodawanie studenta [21].

Po wypełnieniu listy studentów osoba obsługująca system ma możliwość dodania wybranych osób do zlecenia – wydania legitymacji. Aby to zrobić użytkownik musi zaznaczyć odpowiednie pola wyboru znajdujące się po lewej stronie ekranu, a następnie

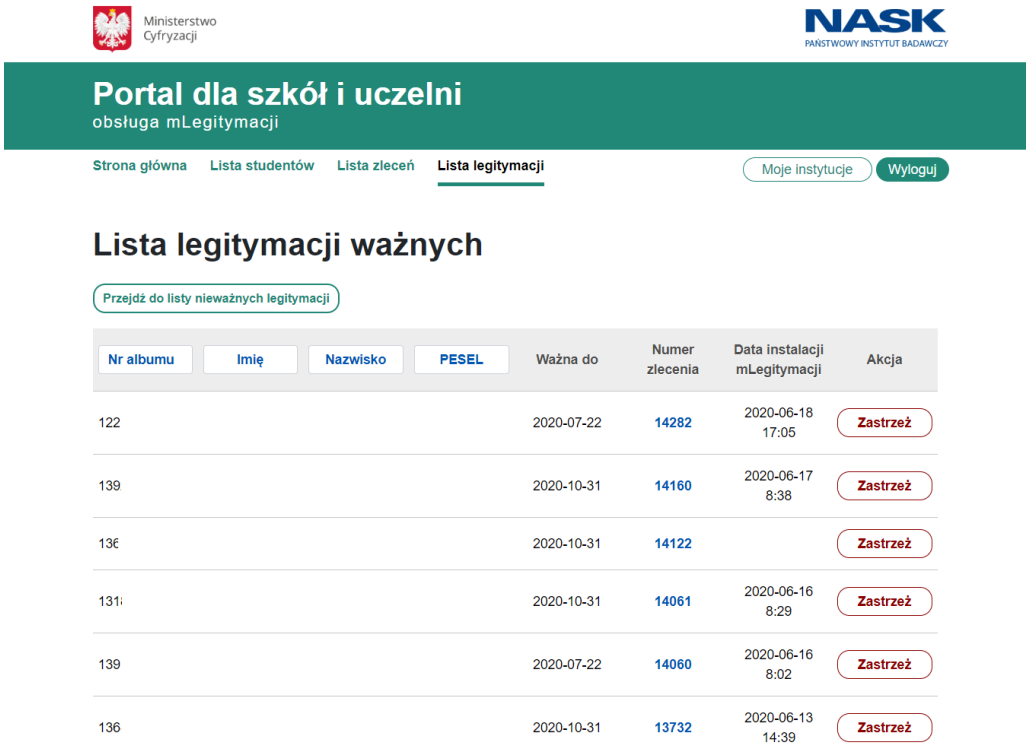
kliknąć przycisk „Utwórz zlecenie”. Wówczas nastąpi automatyczne przeniesienie do zakładki „Lista zleceń”, w której pojawi się właśnie utworzone zlecenie o statusie „Nowy” (rysunek 3.8).



Rysunek 3.8 Zakładka „Lista zleceń” [21].

Dla każdego nowego zlecenia możliwe jest wygenerowanie raportu z listą utworzonych legitymacji. Jednak aby tego dokonać konieczna jest autoryzacja użytkownika Profilem Zaufanym oraz kodem SMS. Po poprawnej autoryzacji zlecenie zostanie przetworzone i zmieni status na „Gotowe do pobrania raportu”. Użytkownik po kliknięciu przycisku „Pobierz raport” będzie miał wgląd do listy wszystkich wygenerowanych legitymacji.

Kolejną procedurą dostępną dla osoby obsługującej system jest zastrzeżenie legitymacji studenckiej. Podobnie jak w przypadku wydania legitymacji wymaga ona najpierw zalogowania się do systemu. Następnie wymagane jest przejście do zakładki „Lista legitymacji” (rysunek 3.9). Użytkownik ma w tym miejscu możliwość wyszukania odpowiedniej legitymacji poprzez wprowadzenie wartości następujących pól: Numer legitymacji, Imię, Nazwisko, PESEL. Dla wybranej ważnej legitymacji możliwe jest wykonanie akcji „Zastrzeż”. Po potwierdzenia tego działania dana legitymacja zostanie unieważniona, a wartość pola status zmieni się na „Nieważna”.



Rysunek 3.9 Zakładka „Lista legitymacji” [21].

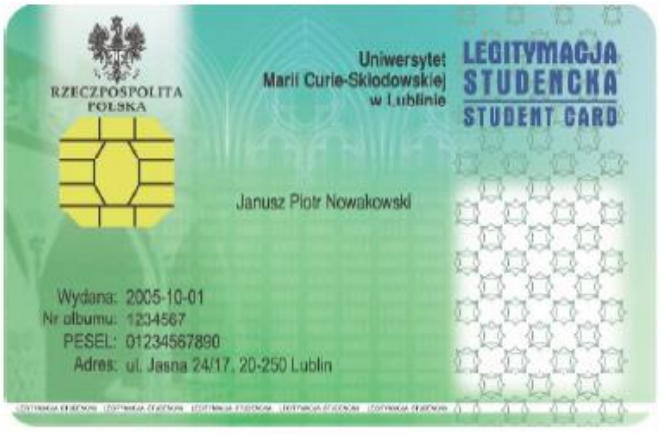
Ostatnią z dostępnych dla użytkownika procedur jest przedłużanie legitymacji studenckich. Procedurę tę należy również rozpocząć od zalogowania się do systemu. Następnie należy przejść do zakładki „Lista studentów” i wybrać jedną z dostępnych opcji – „Dodaj studenta” lub „Importuj dane”. Przy przedłużaniu legitymacji wskazane jest jednak korzystanie z narzędzia do importu danych. Jeśli przedłużone mają zostać legitymacje wydane w roku poprzednim, użytkownik powinien mieć dostęp do folderu z listą studentów oraz z ich zdjęciami. Aktualność danych studentów zapisanych na liście musi zostać zweryfikowana przez osobę obsługującą system. Następnie powinna ona zmienić datę ważności legitymacji, zapisać wprowadzone zmiany oraz wygenerować plik XML, który należy zaimportować do systemu. Po prawidłowym zaimportowaniu danych studentów użytkownik musi utworzyć zlecenie. Tak samo jak w przypadku wydawania legitymacji, zlecenie to zostanie dodane do zakładki „Lista zleceń”, po czym zostanie ono przetworzone. Dla przetworzonego zlecenia możliwe jest wygenerowanie oraz pobranie raportu zawierającego dane studentów i przypisany do każdego z nich kod QR.

3.4 Kwestie bezpieczeństwa związane z wprowadzeniem struktury ELS w wersji 2

Rozporządzenie Ministra Edukacji Narodowej i Sportu z dnia 18 lipca 2005 r. zobowiązuje uczelnię do zapewnienia bezpieczeństwa danych przechowywanych na Elektronicznej Legitymacji Studenckiej [17]. W celu zagwarantowania bezpieczeństwa ELS wykorzystywanych jest kilka poziomów zabezpieczeń [22]. Należą do nich:

- zabezpieczenia graficzne,
- mechanizmy zabezpieczeń w systemie operacyjnym karty oraz w strukturze plików zapewnione przez producenta,
- mechanizmy dotyczące struktury danych (podpis kwalifikowany),
- hologramy,
- system zarządzania cyklem życia karty.

Zastosowanie zabezpieczeń graficznych chroni przed podrabianiem Elektronicznej Legitymacji Studenckiej. Na etapie produkcji offsetowo nanoszone są elementy wizualne: gilosze oraz mikrodruk (rysunek 3.10). Gilosz drukowany jest pionowo, w pasie znajdującym się po prawej stronie legitymacji, który obejmuje również miejsce na zdjęcie studenta. Mikrodruk natomiast umieszczany jest na białym poziomym pasku przechodzącym wzdłuż dolnej części awersu legitymacji. Na pasku tym drukowany jest powtarzający się napis „LEGITYMACJA STUDENCKA".



Rysunek 3.10 Gilosze i mikrodruk na awersie legitymacji studenckiej [22].

Bezpieczne zarządzanie aplikacjami w ELS zapewnione jest poprzez zastosowanie implementacji GlobalPlatform (opisanej w rozdziale 4.2.). Z kolei mechanizmy bezpieczeństwa struktury plików na karcie określone zostały w normie ISO/IEC 7816-4. Dokument ten definiuje pięć stanów zabezpieczeń, atrybuty

określające dostępne operacje i warunki ich wykonania, a także cztery mechanizmy bezpieczeństwa: uwierzytelnianie użytkownika za pomocą klucza, uwierzytelniania użytkownika za pomocą hasła, uwierzytelnianie danych oraz szyfrowanie danych.

Rozporządzenie Ministra wymaga, aby struktura z danymi studenta została podpisana bezpiecznym podpisem elektronicznym weryfikowanym za pomocą certyfikatu kwalifikowanego. Format podpisywanej wiadomości dla ELS w wersji 1 musi być zgodny ze specyfikacją ETSI TS 101 733. Ponieważ jednym z elementów podpisywanej struktury jest numer seryjny układu na karcie, stąd weryfikacja podpisu elektronicznego powinna potwierdzić autentyczność danej legitymacji studenckiej.

Kolejnym elementem zabezpieczeń są hologramy. Służą one do weryfikacji ważności ELS w trybie „off-line". Hologramy są wykonane w taki sposób, który nie pozwala na ich odklejenie, gdyż powoduje to ich zniszczenie.

Ostatni poziom zabezpieczeń związany jest z systemem zarządzania kartami. Zgodnie z rozporządzeniem obowiązek zapewnienia bezpieczeństwa systemu wydawania oraz użytkowania ELS należy do uczelni. System ten pozwala m.in. na zarządzanie hasłami, kluczami oraz aplikacjami znajdującymi się na karcie, dywersyfikację kluczy, a także zarządzanie kolejnymi etapami życia legitymacji, do których należą: personalizacja, wydanie, użytkowanie i zniszczenie.

Elektroniczna Legitymacja Studencka w wersji 2 w zdecydowanej większości opiera się na tych samych założeniach w kwestii bezpieczeństwa. Pierwszą znaczącą różnicą jest zmiana sposobu podpisywania struktury zawierającej dane studenta. Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. określa, że struktura podpisywanej wiadomości musi być zgodna z europejską normą ETSI EN 319 122-1 i musi być ona opatrzona kwalifikowanym podpisem albo kwalifikowaną pieczęcią elektroniczną [18]. Ponieważ w nowej wersji ELS istnieje możliwość dodania pliku EF.PHOTO ze zdjęciem studenta, stąd konieczne było wprowadzenie zabezpieczeń chroniących ten plik przed nieautoryzowanymi zmianami. W tym celu w podpisywanej strukturze z danymi studenta umieszczone zostały pola: `funkcjaSkrotu` oraz `skrotZdjecia`. Ich obecność powoduje, że jakakolwiek zmiana pliku ze zdjęciem wymaga również zmiany w podpisywanej strukturze.

Nowe rozporządzenie Ministra wprowadza w życie możliwość korzystania z mLegitymacji jako pełnoprawnej alternatywy dla karty z ELS. W związku z tym stworzony został specjalny portal do jej obsługi od strony uczelni, a także skojarzona

z nim aplikacja dla użytkowników będąca częścią usługi mObywatel. Oba te systemy muszą zapewniać bezpieczeństwo oraz autentyczność przechowywanych danych.

W tym celu portal do obsługi mLegitymacji wyznacza ścisłe reguły użytkowania dla wszystkich uprawnionych pracowników uczelni [23]. Osoby te mogą korzystać z portalu wyłącznie na komputerach służbowych. Oprócz tego każdy użytkownik obsługujący system musi uwierzytelniać się z wykorzystaniem Profilu Zaufanego zabezpieczonego hasłem, a podczas logowania jest on zobowiązany do sprawdzenia certyfikatu bezpieczeństwa strony.

Usługa mLegitymacja korzysta z tych samych mechanizmów bezpieczeństwa co aplikacja mObywatel [24]. Dane w niej przechowywane zabezpieczone są za pomocą hasła lub odcisku palca ustawianego podczas aktywacji aplikacji. Logowanie przy użyciu odcisku palca jest dodatkowo zabezpieczone PIN-em. Wszystkie dane pobrane i zapisane w telefonie w procesie aktywacji są zaszyfrowane, co uniemożliwia dostęp do nich osobom trzecim. Poza tym w celu ochrony danych osobowych użytkownika do weryfikacji przekazywane są jedynie niezbędne informacje: imię (imiona), nazwisko oraz wiek studenta, numer legitymacji wraz z datą jej wydania i ważności, nazwa i adres uczelni oraz zdjęcie o zmniejszonej jakości wzbogacone o znak wodny. Dodatkowo poza dostępnymi funkcjami przekazania i weryfikacji danych mLegitymacja nie daje możliwości importowania ani eksportowania danych. Nie zezwala ona także na wykonywanie zrzutów ekranu. Podobnie jak w przypadku fizycznej wersji ELS również mLegitymacja wyposażona jest w odpowiednie zabezpieczenia wizualne zapewniające jej autentyczność. Należą do nich:

- tło o powtarzalnym wzorze,
- hologram przedstawiający Godło Rzeczypospolitej Polskiej o zmiennej barwie zależnej od kąta nachylenia telefonu,
- flaga państwowa przedstawiona jako animacja biało-czerwonej flagi powiewającej na wietrze,
- data wydania Elektronicznej Legitymacji Studenckiej,
- data ważności Elektronicznej Legitymacji Studenckiej,
- czas okazania mLegitymacji na telefonie.

4. Technologia Java Card

W rozdziale tym omówiono komponenty technologii Java Card oraz środowiska GlobalPlatform, na których opiera się projekt ELS w wersji 2. Oprócz tego porównano ze sobą możliwości wykonania apletu w różnych wersjach Java Card: Java Card 2.2.2, Java Card 3.0.5 (Classic i Connected) oraz Java Card 3.1.0.

4.1 Budowa Java Card OS

Elektroniczna Legitymacja Studencka, której nowa wersja jest przedmiotem pracy, działa w oparciu o technologię Java Card. Na tę technologię składają się następujące komponenty: maszyna wirtualna (Java Card VM), środowisko uruchomieniowe (Java Card RE) oraz interfejs (Java Card API), które zostaną omówione w kolejnych podrozdziałach.

4.1.1 Maszyna wirtualna Java Card VM

Specyfikacja definiuje wirtualną maszynę Javy jako maszynę, która ładuje pliki klasy Java i wykonuje je z określonym zestawem semantyki [25]. Pełna implementacja wirtualnej maszyny Java jest jednak zbyt duża, aby zmieściła się na urządzeniach o ograniczonych zasobach pamięci, jakimi są karty elektroniczne. Stąd maszyna wirtualna Java Card stanowi zawężony podzbiór maszyny wirtualnej Java. Java Card VM nie posiada zatem wielu elementów, które są zaimplementowane w normalnej maszynie wirtualnej Java. Należą do nich:

- dynamiczne ładowanie klas - programy działające na karcie mogą odnosić się tylko do klas, które już istnieją na karcie (np. zostały zapisane na niej podczas produkcji), ponieważ nie ma możliwości pobrania klas podczas normalnego wykonywania kodu aplikacji;
- Security Manager - na platformie Java Card funkcje bezpieczeństwa języka są implementowane przez maszynę wirtualną, ponieważ nie posiada ona klasy Security Manager, która podejmowałaby decyzje dotyczące tego, czy zezwalać na operacje;
- finalizacja - maszyna wirtualna Java Card nie wywołuje automatycznie funkcji finalize();
- wielowątkowość - programy Java Card nie mogą korzystać z klasy Thread ani z żadnych ze słów kluczowych związanych z wątkami, dostępnych w języku Java;

- klonowanie obiektów - platforma Java Card nie obsługuje klonowania obiektów;
- typy enum - maszyna wirtualna Java Card nie obsługuje typu wyliczeniowego;
- ulepszona pętla for - Java Card obsługuje tylko indeksowanie tablic przy użyciu typu short, ale nie obsługuje ulepszonej konstrukcji pętli for, która wymaga obsługi indeksowania tablic przy użyciu typu integer;
- adnotacje - platforma Java Card nie obsługuje mechanizmów refleksji;
- asercje - środowisko wykonawcze Java Card nie zapewnia obsługi asercji, które są używane do testowania założeń dotyczących funkcjonalności programu.

Java Card VM nie obsługuje również następujących typów zmiennych:

- char,
- double,
- float,
- long,
- tablice wielowymiarowe.

Do elementów zaimplementowanych w Java Card VM należą z kolei:

- pakiety - klasy Java Card API są zapisywane jako pliki źródłowe Java, które zawierają oznaczenia pakietów, a mechanizmy pakietów służą do identyfikacji i kontroli dostępu do klas, pól statycznych i metod statycznych;
- dynamiczne tworzenie obiektów - programy Java Card obsługują dynamicznie tworzone obiekty, zarówno instancje klas, jak i tablice;
- metody wirtualne - wywoływanie metod wirtualnych na obiektach w programie napisanym dla platformy Java Card jest dokładnie takie samo, jak w programie napisanym dla platformy Java, z obsługą dziedziczenia, w tym z użyciem słowa kluczowego super;
- interfejsy - klasy Java Card API mogą definiować lub implementować interfejsy;
- wyjątki - programy Java Card mogą definiować, wyrzucać oraz łapać wyjątki, obsługiwana jest klasa Throwable oraz jej odpowiednie podklasy, ale niektóre podklasy wyjątków i błędów zostały w Java Card pominięte;
- struktury generyczne - język Java Card pozwala typom lub metodom operować na obiektach różnych typów przy jednoczesnym zapewnieniu bezpieczeństwa typów w trakcie kompilacji, wyeliminowana jest również potrzeba rzutowania;

- import statyczny - pozwala on uniknąć importowania całej klasy w celu uzyskania dostępu do jej elementów statycznych.

Java Card VM obsługuje także następujące typy zmiennych:

- boolean,
- byte,
- short,
- int (opcjonalnie),
- obiekty (instancje klas i tablice jednowymiarowe).

Jedną z najistotniejszych różnic pomiędzy maszyną wirtualną Javy, a maszyną wirtualną Java Card jest okres jej życia [26]. Na komputerze PC lub stacji roboczej VM Java działa jako proces systemu operacyjnego. Po jego zakończeniu aplikacje w języku Java i ich obiekty są automatycznie niszczone. W technologii Java Card czas życia maszyny wirtualnej jest czasem życia karty. Karta po włożeniu do czytnika jest zasilania i Java Card VM zaczyna działać. Po wyłączeniu zasilania, maszyna wirtualna zatrzymuje się tylko tymczasowo. Gdy karta zostanie następnie zresetowana, VM uruchomia się ponownie i odzyskuje poprzedni stos obiektów z pamięci trwałej. Technologia trwałej pamięci (np. EEPROM) sprawia, że większość informacji przechowywanych na karcie zostaje zachowana nawet po odłączeniu zasilania z karty. Ponieważ zarówno maszyna wirtualna, jak i obiekty utworzone na karcie służą do reprezentowania trwałych informacji aplikacji, VM Java Card zdaje się działać nieustannie.

4.1.2 Środowisko uruchomieniowe Java Card RE

Java Card Runtime Environment stanowi zestaw elementów niezbędnych do uruchomienia apletów napisanych w Java Card [26]. Środowisko to odpowiada za wiele aspektów pracy: nadzoruje cykl życia apletu, obsługuje kanały logiczne, zapewnia bezpieczeństwo i atomowość przeprowadzanych operacji oraz dostarcza wiele funkcji i możliwości, z których aplet może korzystać.

Cykl życia instancji apletu rozpoczyna się, gdy zostanie on pomyślnie zarejestrowany w Java Card RE za pomocą metody **Applet.register**. Aplety zarejestrowane za pomocą tej metody istnieją aż do momentu usunięcia ich przez Menadżera usuwania apletów.

Java Card RE inicjuje interakcje z apletem za pomocą publicznych metod apletu:

- **install –** głównym zadaniem tej metody jest utworzenie nowej instancji apletu za pomocą jej konstruktora oraz zarejestrowanie tej instancji;
- **select –** metoda ta pozwala na wybranie apletu, dzięki czemu kolejne komendy APDU będą przesyłane bezpośrednio do metody *process* tego apletu;
- **process –** służy do interpretacji komend APDU oraz wykonania dla nich odpowiednich instrukcji przez aplet;
- **deselect –** metoda ta dezaktywuje aktualnie wybrany aplet. Umożliwia jednak wykonanie pewnych operacji przed zakończeniem aktualnej sesji;
- **uninstall –** metoda ta jest wywoływana przed usunięciem apletu z karty.

Java Card w wersji 3 posiada również wiele innych przydatnych funkcji. Jedną z nich jest możliwość obsługi kanałów logicznych. Pozwala ona na otwarcie maksymalnie 20 połączeń poprzez dowolny terminal wejścia/wyjścia (I/O). Funkcjonalność kanałów logicznych umożliwia m.in. obsługę wielu sesji, równoległy wybór kilku apletów lub kilkukrotny wybór jednego apletu na różnych kanałach jednocześnie.

Inną dostępną funkcją w Java Card RE są obiekty tymczasowe (*ang. transient objects*). Mogą one służyć do przechowywania danych potrzebnych jedynie przez określony czas, bowiem ich dane są czyszczone natychmiast po zakończeniu pracy z apletem (spowodowanym np. poprzez wybór innego apletu). Z powodów bezpieczeństwa obiekty tymczasowe nie mogą być przechowywane w pamięci trwałej. Przy obecnej architekturze kart oznacza to, że mogą znajdować się one w pamięci RAM, natomiast nie w pamięci EEPROM. Taki sposób zapisu obiektów tymczasowych pozwala na ich wykorzystanie do przechowywania kluczy sesji.

Środowisko uruchomieniowe zapewnia również izolację apletów za pomocą mechanizmu kontekstu. Izolacja oznacza, że jeden aplet nie może uzyskać dostępu do pól lub obiektów apletu znajdującego się w innym kontekście. Ochronę przed wyciekiem poufnych danych z apletu zapewnia zapora ogniowa Java Card, która odrzuca żądania innego apletu niespełniającego określonych reguł dostępu. Istnieje jednak możliwość współdzielenia obiektów między apletami w różnych kontekstach za pomocą interfejsu **Shareable.** Interfejs ten definiuje zestaw wspólnych metod, które można wywoływać z jednego kontekstu, nawet jeśli obiekt implementujący je jest własnością apletu w innym kontekście. Pozostałe metody, które nie wchodzą w skład interfejsu nadal będą chronione przez firewall.

Java Card RE zapewnia również wsparcie dla transakcji atomowych, dzięki czemu dane karty są przywracane do pierwotnego stanu przed transakcją, jeśli nie zakończy się ona w normalny sposób. Mechanizm ten chroni zatem przed zdarzeniami, takimi jak utrata zasilania w trakcie transakcji oraz przed błędami programu, które mogą spowodować uszkodzenie danych, jeśli wszystkie etapy transakcji nie zakończą się prawidłowo.

Opcjonalnym składnikiem platformy Java Card 3 jest funkcja wywoływania metod zdalnych *(ang. RMI – Remote Method Invocation)*. Java Card RMI oddaje do dyspozycji użytkownika mechanizm zdalnego wywoływania funkcji na zdalnym obiekcie znajdującym się na karcie. Zdalny obiekt musi implementować co najmniej jeden zdalny interfejs, tj. interfejs dziedziczący po *java.rmi.Remote*. Metody występujące w takim interfejsie są określane jako metody zdalne.

4.1.3 Interfejs API Java Card Application Programming Interface

Ostatnim komponentem składającym się na technologię Java Card jest Java Card API. Stanowi on zbiór pakietów i elementów wykorzystywanych przy programowaniu kart elektronicznych z wykorzystaniem tej technologii. Do najczęściej używanych pakietów języka Java należą [27]:

java.io – definiuje podzbiór pakietu java.io w standardowym języku Java związany z wyjątkami (IOException);

java.lang – zapewnia podstawowe klasy języka Java wykorzystywane przy programowaniu kart w technologii Java Card (udostępnia klasy Object i Throwable oraz klasy wyjątków);

java.rmi – definiuje interfejs Remote, metody interfejsów, które go rozszerzają można wywoływać z poziomu aplikacji klienckich urządzeń akceptujących karty (*ang. CAD – Card Acceptance Device*);

javacard.framework – dostarcza strukturę klas i interfejsów potrzebnych do budowania, komunikacji i pracy z apletami opartymi na technologii Java Card;

javacard.security – zapewnia klasy i interfejsy zawierające publiczne funkcje bezpieczeństwa i kryptografii platformy Java;

javacardx.annotations – pakiet rozszerzeń zawierający adnotacje do definiowania stałych ciągów znaków;

javacardx.apdu – pakiet rozszerzeń, który umożliwia obsługę zdefiniowanych przez normę ISO7816 opcjonalnych mechanizmów powiązanych z APDU;

javacardx.apdu.util – pakiet rozszerzeń z klasą APDUUtil, która zawiera funkcje przetwarzające CLA z komendy APDU;

javacardx.biometry – pakiet rozszerzeń, który zawiera funkcjonalność do implementacji struktury biometrycznej na platformie Java Card;

javacardx.biometry1toN – pakiet rozszerzeń, który zawiera funkcjonalność do implementacji struktury biometrycznej 1:N na platformie Java Card;

javacardx.crypto – pakiet rozszerzeń zawierający dodatkowe funkcje kryptograficzne.

javacardx.external – pakiet rozszerzeń zapewniający mechanizmy dostępu do podzespołów pamięci, które nie mogą być zaadresowane bezpośrednio przez Java Card Runtime Environment;

javacardx.framework.math – pakiet rozszerzeń zawierający typowe funkcje narzędziowe do obliczeń kodowania BCD i parzystości;

javacardx.framework.string – pakiet rozszerzeń, który zawiera typowe funkcje narzędziowe do manipulowania ciągami znaków zakodowanych w UTF-8;

javacardx.framework.util – pakiet rozszerzeń zawierający funkcje narzędziowe do manipulowania tablicami typów prostych – byte, short, int;

javacardx.framework.util.intx – pakiet rozszerzeń z funkcjami narzędziowymi do obsługi typu int;

javacardx.security – pakiet rozszerzeń zawierający funkcjonalność pozwalającą na implementację środków bezpieczeństwa w celu ochrony istotnych apletów Java Card.

4.2 Środowisko GlobalPlatform

Aby karty elektroniczne mogły osiągnąć swój prawdziwy potencjał, konsumenci muszą mieć możliwość korzystania z nich w szerokim zakresie funkcji [28]. Na przykład można ich używać z telefonami komórkowymi do robienia zakupów przez Internet, a także w celu bezpiecznego dostępu do komputera. By zmniejszyć bariery utrudniające rozwój międzybranżowych kart elektronicznych do wielu zastosowań, organizacja GlobalPlatform zdefiniowała elastyczną i wydajną specyfikację wspólną dla wszystkich wydawców kart elektronicznych. Specyfikacja ta pozwala im wybrać technologię, której obecnie potrzebują, jednocześnie zapewniając możliwość ewentualnej migracji na inną technologię w przyszłości, bez znaczących zmian w infrastrukturze karty.

Architektura kart zgodnych z GlobalPlatform złożona jest z następujących komponentów:

- Security Domains (SD),
- Global Services Applications,
- Runtime Environment,
- Trusted Framework,
- GlobalPlatform Environment (wcześniej Open Platform Environment, OPEN),
- GlobalPlatform API,
- Zawartość karty,
- Card Manager.

Security Domains to specjalne aplikacje do zarządzania bezpieczeństwem i kluczami szyfrującymi. Ich celem jest zapewnienie separacji kluczy szyfrujących pomiędzy wystawcą karty oraz pozostałymi aplikacjami typu Security Domain. Oprócz tego odpowiadają one również za szyfrowanie, deszyfrowanie, generowanie i weryfikację podpisów cyfrowych dla aplikacji ich dostawców (wystawcy karty, dostawcy aplikacji lub organu kontrolnego).

Na karcie mogą znajdować się także **aplikacje Global Services**, których zadaniem jest dostarczenie usług pozostałym aplikacjom znajdującym się na karcie. Przykładem takiej usługi może być metoda weryfikacji posiadacza karty (*ang. Cardholder Verification Method*), która udostępnia globalny PIN.

GlobalPlatform jest przeznaczone do działania na dowolnym, bezpiecznym środowisku wykonawczym obsługującym wiele aplikacji. To środowisko uruchomieniowe (**Runtime Environment**) jest odpowiedzialne za zapewnienie niezależnego od sprzętu API dla aplikacji, a także bezpiecznego miejsca do przechowywania i wykonywania aplikacji, aby zarówno kod jak i dane różnych aplikacji zostały bezpiecznie od siebie odseparowane.

GlobalPlatform może również zawierać zaufane struktury (**Trusted Framework**), które zapewniają usługi komunikacji między aplikacjami. Stanowią one część środowiska uruchomieniowego karty, mimo że nie są ani aplikacjami ani SD.

Do głównych zadań **GlobalPlatform Environment (OPEN)** należy zapewnienie API dla aplikacji, wysyłanie poleceń, wybór aplikacji, zarządzanie kanałem logicznym (opcjonalne) oraz zarządzanie zawartością karty. Funkcje te zostaną zaimplementowane przez OPEN, jeśli środowisko wykonawcze ich nie zapewnia lub jeśli są one zapewnione przez środowisko wykonawcze w sposób niezgodny ze specyfikacją GlobalPlatform.

GlobalPlatform API zapewnia kilka dodatkowych funkcji wykorzystywanych przez aplikacje GlobalPlatform oraz przez Card Managera. Do funkcji tych należą m.in. weryfikacja posiadacza karty, blokada karty w przypadku stwierdzenia, że bezpieczeństwo karty mogło zostać naruszone oraz sprawdzanie wartości kontrolnej klucza przed załadowaniem klucza na kartę.

Zawartość karty jest dostępna w postaci wykonywalnego pliku ładowania, który może znajdować się:

- w stałej pamięci trwałej - wtedy jest tylko ładowany i nie może być zmieniany,
- w zmiennej pamięci trwałej - wtedy może być zarówno załadowany jak i usunięty.

Każdy wykonywalny plik ładowania może zawierać jeden lub wiele modułów wykonywalnych, będących kodem aplikacji. Karta GlobalPlatform jest przeznaczona do obsługi wielu wykonywalnych plików ładowania oraz wielu modułów wykonywalnych, dzięki czemu możliwe jest współistnienie na niej wielu aplikacji.

GlobalPlatform Card Manager jest jednostką zarządzającą karty elektronicznej. Łączy ona w sobie trzy wcześniej opisane funkcje: GlobalPlatform Environment (OPEN), Security Domain wystawcy karty oraz usługę Cardholder Verification Method.

4.3 Porównanie możliwości wykonania apletu w Java Card 2.2.2, Java Card 3.0.5 oraz Java Card 3.1.0

Aplet Elektronicznej Legitymacji Studenckiej w wersji 1 został napisany w oparciu o Java Card 2.2.2. Stworzenie apletu w wersji 2 jest idealnym momentem na wykorzystanie bardziej aktualnej technologii Java Card. W ramach projektu ELS2 zdecydowałam się na użycie jednej z nowszych dostępnych wersji, Java Card 3.0.5 Classic Edition. Dodaje ona nowe usprawnienia w działaniu względem wcześniej użytej wersji. Zostaną one omówione w tym rozdziale.

4.3.1 Porównanie wersji 2.2.2 i 3.0.5

Java Card 3.0.5 Classic wydana w czerwcu 2015 r. jest tak naprawdę ewolucją Java Card 2.2.2. Posiada ona dostęp do wszystkich funkcji dostępnych we wcześniejszych wersjach, dzięki czemu aplety napisane w wersji 2.2.2 mogą być z powodzeniem uruchamiane w ramach wersji 3.0.5. Jednocześnie Java Card 3.0.5 Classic została wzbogacona o całkiem nowe funkcjonalności, które znacząco zwiększają możliwości tej technologii.

Najważniejsze zmiany w wersji 3.0.5 dotyczą kwestii bezpieczeństwa. Wprowadzone zostały rozszerzone możliwości obsługi PIN-u, niezwykle przydatne w sektorze bankowości [29]. Do pakietu **javacard.framework** dodana została nowa klasa OwnerPINBuilder służąca do tworzenia nowych obiektów OwnerPIN. Oprócz tego dodane zostały nowe interfejsy – OwnerPINx i OwnerPINxwithPredecrement. Pierwszy z nich reprezentuje numer PIN właściciela i rozszerza jego funkcjonalność poprzez zapewnienie możliwości aktualizacji kodu PIN, aktualizacji limitu prób oraz użycie licznika prób. OwnerPINxwithPredecrement jest z kolei rozszerzeniem interfejsu OwnerPINx. Odpowiada on za zmniejszanie licznika prób przed jakąkolwiek próbą walidacji kodu PIN.

Kolejną zmianą w Java Card 3.0.5 jest dołączenie do pakietu javacard.framework klasy SensitiveArrays. Klasa ta zapewnia metody tworzenia i obsługi obiektów tablicowych wrażliwych na integralność. Dzięki temu rozwiązaniu znacznie uproszczona zostaje ochrona integralności danych przed zewnętrznymi atakami. Ponieważ równie ważne jest zapewnienie atomowości operacji na danych przechowywanych w tablicach, do pakietu javacard.framework.Util dodana została nowa metoda arrayFill, która wypełnia tablice w sposób atomowy.

Nowe ulepszenia w kwestii bezpieczeństwa osiągnięto poprzez wprowadzenie zmian w pakiecie **javacard.security**. Dodano do niego interfejsy DHKey, DHPrivateKey oraz DHPublicKey związane z kluczami stosowanymi w algorytmie Diffiego-Hellmana. W celu zapewnienia dodatkowego wsparcia ochrony danych domenowych dla algorytmów Diffiego-Hellmana, krzywych eliptycznych (ang. *Elliptic Curve*) oraz DSA, w klasie KeyBuilder zaimplementowane zostały stałe ALG_TYPE_DH_PARAMETERS, ALG_TYPE_DSA_PARAMETERS, ALG_TYPE_EC_F2M_PARAMETERS, ALG_TYPE_EC_FP_PARAMETERS oraz metoda buildKeyWithSharedDomain(). Znaczące zmiany dokonane zostały również w klasie Signature. Dodana została w niej klasa rozszerzająca OneShot, umożliwiająca obsługę operacji podpisywania i weryfikacji za jednym razem, dzięki czemu mogą one całkowicie uniknąć zapisów w pamięci trwałej. W klasie Signature pojawiła się również nowa metoda verifyPreComputedHash(), służąca do weryfikacji wstępnie obliczonego skrótu oraz zdefiniowane zostały nowe stałe stosowane w algorytmach podpisu cyfrowego: ECDSA oraz AES CMAC. Aby umożliwić także obsługę kryptograficznej funkcji skrótu SHA-3, w klasie MessageDigest dodane zostały odpowiednie stałe –

ALG_SHA3_224, ALG_SHA3_256, ALG_SHA3_384, ALG_SHA3_512, LENGTH_SHA3_224, LENGTH_SHA3_256, LENGTH_SHA3_384 oraz LENGTH_SHA3_512.

Istotne zmiany w wersji 3.0.5 Java Card dotyczą również kwestii kryptografii. W klasie Cipher pakietu **javacardx.crypto** zaimplementowane zostały: nowa stała ALG_AES_CTR do obsługi szyfru używającego AES w trybie CTR (ang. *Counter*), nowe stałe do obsługi SHA-3 oraz nowe stałe określające sposób wyrównywania danych według schematu PKCS#1-OAEP dla algorytmów SHA224, SHA256, SHA384 i SHA512. Oprócz tego klasa Cipher jest rozszerzana przez klasę OneShot, przeznaczoną do obsługi operacji szyfrowania i deszyfrowania za jednym razem. Nowością w pakiecie javacard.crypto jest także klasa AEDCipher, będąca abstrakcyjną klasą bazową dla szyfrów uwierzytelniających AEAD.

W nowej wersji Java Card poza wyżej wymienionymi zmianami zostało dodanych jeszcze kilka pakietów. Są to:

- **javacardx.apdu.util**, który zawiera klasę APDUUtil z użytecznymi funkcjami do analizy bajta CLA z polecenia APDU;
- **javacardx.biometry1toN**, który zawiera funkcjonalność do implementacji struktury biometrycznej 1:N na platformie Java Card;
- **javacardx.security**, który zawiera funkcje do wdrażania środków bezpieczeństwa w celu ochrony istotnych zasobów apletów; zawiera on klasę SensitiveResult, która zapewnia metody sprawdzające wyniki funkcji wrażliwych;
- **javacardx.framework.util**, w którym do klasy ArrayLogic dodane zostały nowe metody arrayFillGeneric() oraz arrayFillGenericNonAtomic().

4.3.2 Porównanie wersji 3.0.5 Classic i Connected

Od wersji 3.0 platforma Java Card jest udostępniana w dwóch edycjach – Classic oraz Connected. Obie z nich są wstecznie kompatybilne z poprzednimi wersjami Java Card oraz współdzielą kluczowe funkcje bezpieczeństwa.

Java Card 3.0.5 w edycji **Classic** jest ewolucją architektury platformy Java Card 2.2.2 [30]. Podobnie jak w przypadku poprzednich wersji tej platformy, opiera się ona na technologii 16-bitowej dzielonej maszyny wirtualnej, która umożliwia wstępne przetwarzanie aplikacji, które zostaną załadowane na kartę. Ta technologia maszyny

wirtualnej zapewnia, że Java Card może zostać zaimplementowana na kartach z minimalnymi wymaganiami względem pamięci i procesora. I chociaż Java Card 3.0.5 Classic nie różni się wiele pod względem funkcjonalnym od wersji wcześniejszych, znacząco usprawnia ona niektóre aspekty pracy z aplikacjami bazującymi na APDU.

Prawdziwą rewolucją w świecie programowania kart elektronicznych jest natomiast „właściwa" nowa wersja, czyli Java Card **Connected**. W ogromnym stopniu rozszerza ona możliwości programowania najnowszych kart elektronicznych. Java Card 3.0.5 Connected zapewnia wysokiej jakości kartom elektronicznym lepszą łączność oraz integrację z sieciami all-IP (*ang. Next Generation Network*), a także pozwala ona im pełnić rolę bezpiecznego węzła sieci, świadczącego usługi bezpieczeństwa sieci. Wersja ta wprowadza również możliwość zbiegania się wielu usług kart w ramach równoległej komunikacji poprzez interfejsy kontaktowe (protokół IP i protokoły zgodne z ISO 7816-4) oraz interfejsy bezkontaktowe (protokoły zgodne z ISO 14443).

Oprócz tego technologia Java Card 3.0.5 Connected oferuje znacznie ulepszone środowisko wykonawcze oraz nową 32-bitową maszynę wirtualną. Maszyna ta bazuje na maszynie wirtualnej o konfiguracji CLCD (*ang. Connected Limited Device Configuration*), która jest powszechnie używana w telefonach komórkowych. Dzięki temu wspiera ona niektóre bardziej zaawansowane funkcje języka Java. Maszyna wirtualna oparta na CLCD została również odpowiednio ulepszona, tak aby mogła spełniać wymagania bezpieczeństwa: jej rozmiar został zredukowany, dodano w niej obsługę protokołów kart elektronicznych oraz zabezpieczenia. Podczas gdy wersja Classic wykorzystywała tradycyjną technikę dzielonej maszyny wirtualnej, w której funkcje ładowania, łączenia i weryfikacji maszyny były wykonywane przez narzędzie konwersji poza kartą, maszyna wirtualna w wersji Connected jest w stanie bezpośrednio ładować pliki klas.

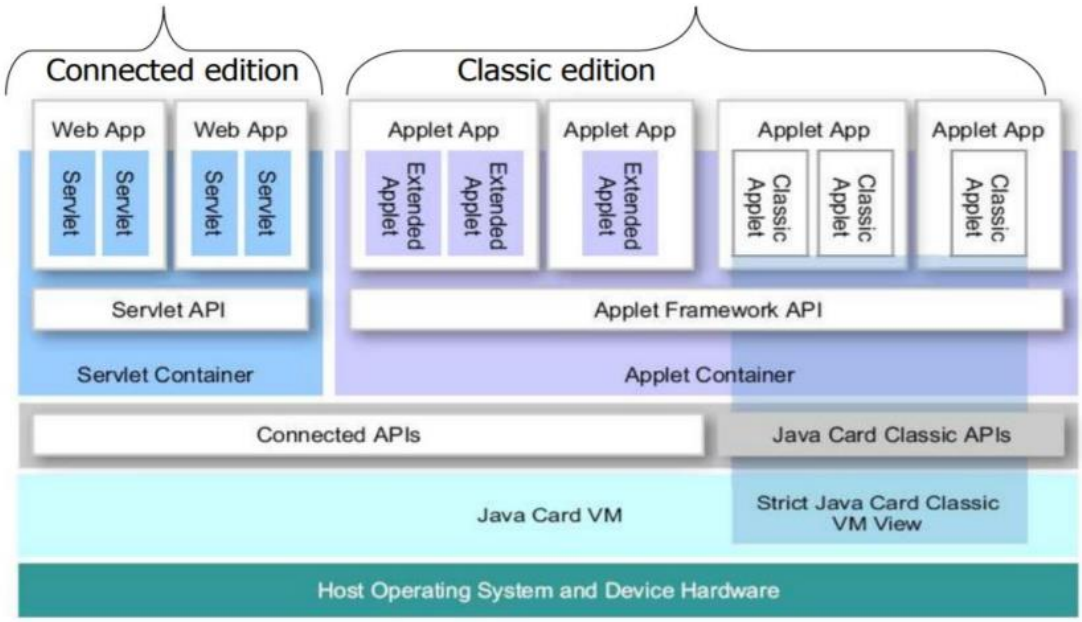
Kolejnym usprawnieniem, jakie dostarcza Java Card Connected, jest wielowątkowość. Aplikacje mogą tworzyć nowe wątki pracujące w tle, jak również przetwarzać równolegle wiadomości. Oprócz tego wersja Connected umożliwia weryfikację kodu bajtowego na karcie. Pliki klas aplikacji są sprawdzane pod kątem bezpieczeństwa typu przez maszynę wirtualną. W celu wydajnej weryfikacji kodu aplikacji przy użyciu ograniczonej ilości pamięci ulotnej dostępnej na karcie, wykorzystywane są informacje o atrybutach mapy stosu w plikach klas generowanych przez kompilator Java SE Development Kit 1.6. Wersja Connected, w odróżnieniu od

Classic, pozwala również na „odśmiecanie pamięci" (*ang. automatic garbage collection*). Tymczasowe dane sesji zostają automatycznie usuwane, gdy nie są już używane.

Aby stworzyć bardziej przyjazne środowisko pracy dla programistów kart elektronicznych, nowa maszyna wirtualna Java Card 3.05 Connected została wzbogacona o dodatkowe elementy pochodzące z JVM. Należą do nich:

- typy danych: char, long, string;
- tablice wielowymiarowe;
- klasy wrappera dla typów prymitywnych: Boolean, Integer itd.;
- klasy manipulujące łańcuchami znaków: StringTokenizer, StringBuffer, StringBuilder;
- klasy obsługi wielowątkowości: Thread itd.;
- klasy I/O: Reader, Writer, Stream;
- klasy sieciowe z Generic Connection Framework: Connector, Connection itd.;
- klasy kolekcji: Vector, Hashtable, Stack, Iterator itd.;
- klasy daty i czasu: Calendar, Date, TimeZone;
- klasy obsługi lokalizacji i internacjonalizacji: Locale, ResourceBundle itd.

Jedną z największych nowości wprowadzonych w ramach wersji Connected jest obsługa aplikacji sieciowych. Aplikacje te współdziałają z klientami internetowymi poza kartami za pośrednictwem żądań i odpowiedzi HTTP lub HTTPS. Cykl życia aplikacji sieciowych, usługi sieciowe, za pośrednictwem których wysyłane są żądania i odpowiedzi oraz bezpieczeństwo dostępu do tych aplikacji i ich zasobów są zarządzane przez kontener aplikacji internetowych platformy Java Card (rysunek 4.1). Typowa aplikacja sieciowa dla Java Card Connected składa się z : serwletów, filtrów żądań i odpowiedzi, detektorów zdarzeń cyklu życia, zasobów statycznych oraz z deskryptora wdrożenia aplikacji sieciowych. Serwlety i filtry są komponentami aplikacji, które generują dynamiczną treść, zdefiniowaną jako treść obliczana na żądanie klientów i wysyłana z powrotem do klientów w ramach odpowiedzi. Zasoby statyczne, takie jak dokumenty HTML i osadzone obrazy lub obiekty, stanowią z kolei treść statyczną, czyli treść znajdującą się w plikach. Deskryptor wdrożenia aplikacji sieciowych opisuje natomiast komponenty aplikacji, sposób w jaki są one mapowane na żądania klientów oraz ich wymagania bezpieczeństwa (w tym uwierzytelnianie użytkownika i autoryzację, a także wymagania dotyczące bezpiecznej komunikacji.)



Rysunek 4.1 Architektura Java Card 3.05 wersja: Classic i Connected [7].

Innym usprawnieniem dodanym do Java Card Connected są nowe mechanizmy komunikacji pomiędzy aplikacjami, a mianowicie serwisy i zdarzenia. Aplikacja może wystawić w rejestrze centralnym serwis, który będzie mógł być używany przez inne aplikacje. Funkcja ta rozszerza mechanizm interfejsu Shareable z wersji Classic i pozwala wszystkim modelom aplikacji (apletom i aplikacjom internetowym) na korzystanie z serwisu w sposób jednolity. Za pomocą rejestru centralnego aplikacje mogą również powiadamiać się nawzajem o zajściu określonego warunku. Gdy warunek zostanie spełniony, zostaje on zawarty w obiekcie rozszerzającym interfejs Shareable, zwanym zdarzeniem. Zdarzenie to jest następnie przekazywane do detektora zdarzeń, który nasłuchuje na spełnienie warunku. Funkcjonalność ta pozwala aplikacjom na komunikację asynchroniczną. Dodatkowo specyfikacja Java Card Connected definiuje zbiór standardowych serwisów m.in. do uwierzytelniania użytkownika oraz zbiór standardowych zdarzeń dotyczących m.in. resynchronizacji zegara lub zarządzania cyklem życia aplikacji.

4.3.3 Porównanie wersji 3.0.5 i 3.1.0

Najnowszą dostępną wersją platformy Java Card jest wydana 16 stycznia 2019 r. Java Card 3.1.0. Wprowadza ona kolejne znaczące usprawnienia w stosunku do wersji 3.0.5, szczególnie w zakresie bezpieczeństwa, a także w postaci nowych funkcjonalności.

Maszyna wirtualna Java Card 3.1.0 wzbogacona została o obsługę rozszerzonego formatu plików CAP (*ang. Extended CAP files*) [31]. Format ten pozwala plikom CAP na obsługę aplikacji większych niż 64 KB oraz aplikacji zawierających wiele pakietów Java, które są albo prywatne i dostępne tylko dla kodu w pliku CAP, albo eksportowane jako biblioteka współdzielona i dostępne z innych plików CAP. Pakiety prywatne w pliku apletu CAP mogą zawierać statycznie zainicjowane tablice. Format pliku CAP obsługuje również dodatkowy komponent COMPONENT_Static_Resources do przechowywania zasobów statycznych. Komponent ten jest obecny w pliku CAP, gdy do konwersji dodawane są pliki zasobów statycznych.

Inną funkcjonalnością dodaną do najnowszej wersji platformy Java Card jest obsługa widoków tablic (*ang. array views*), czyli widoków istniejących tablic, które pokazują wycinki tablic potencjalnie tylko do odczytu [32]. Za ich pomocą części tablicy mogą być współużytkowane z innym apletem bez udzielania dostępu do całej tablicy. Dzięki temu nie jest już potrzebne dokonywanie kopii zapasowych tablic, w celu współdzielenia danych, co pozwala zaoszczędzić na karcie znaczne ilości pamięci.

Jedną z największych nowości w zakresie bezpieczeństwa jest dodanie do Java Card API 3.1.0 bezpośredniej obsługi certyfikatów [31]. Co prawda poprzednie wersje Java Card wspierały obsługę kluczy, szyfrowania, deszyfrowania, wymiany kluczy oraz podpisów cyfrowych, jednak to w kwestii programistów leżała implementacja obsługi certyfikatów. Dzięki dodaniu pakietu javacardx.security.cert do wersji 3.1.0 Java Card możliwa jest analiza, przechowywanie i zarządzanie certyfikatami zakodowanymi w X.509 DER.

Do Java Card API 3.1.0 dodane zostało także wsparcie dla obsługi dodatkowych trybów szyfrowania AES: AES-CFB oraz AES-XTS. Tryb AES-CFB przeznaczony jest do szyfrowania strumieniowego, z kolei AES-XTS używany jest do ochrony danych przechowywanych w pamięci zewnętrznej. Ponieważ tryb AES-XTS wykorzystuje wartość klucza AES jako dwa podklucze, w związku z tym klucz AES musi mieć podwójną długość (256-bitów, aby wykonać 128-bitowy AES-XTS oraz 512-bitów, aby wykonać 256-bitowy AES-XTS).

Kolejnym udogodnieniem dotyczącym szyfrowania wprowadzonym w ramach wersji 3.1.0 jest udostępnienie programistom możliwości konfiguracji parametrów asynchronicznej generacji kluczy. Nowa metoda dodana do pakietu javacard.security –

genKeyPair() pozwala na generowanie klucza oraz obsługuje obiekt konfiguracyjny dostarczany przez aplikację. Z kolei nowe interfejsy AlgorithmParameterSpec, PrimalityTestParameterSpec w pakiecie javacard.security oraz javacardx.security.derivation.KDFCounterModeSpec odpowiadają za kontrolę parametrów testu pierwszości (na przykład rodzaj testu lub liczbę rund) oraz kontrolę algorytmu generowania liczb losowych i deterministycznego generowania klucza poufnego.

Dodatkowo do najnowszej wersji Java Card dodany został opcjonalny pakiet javacardx.security.util.MonotonicCounter, który umożliwia aplikacji korzystanie z bezpiecznej implementacji liczników monotonicznych [31][32]. Klasa MonotonicCounter pozwala na tworzenie i inicjowanie monotonicznych liczników o różnych rozmiarach (do ośmiu bajtów) oraz na ich bezpieczne zwiększanie, porównywanie i pobieranie z nich wartości. I chociaż wszystkie operacje obiektu klasy MonotonicCounter są atomowe, nie bierze on udziału w transakcjach. Każda aktualizacja obiektu odbywa się w ramach własnej transakcji, która jest zatwierdzana natychmiast, niezależnie od tego, czy normalna transakcja już trwa. Takie zachowanie jest konieczne ze względów bezpieczeństwa i ma miejsce także w klasie OwnerPin.

Innym udogodnieniem w Java Card 3.1.0 jest obecność dwóch pakietów: javacardx.framework.event oraz javacardx.framework.nio, które zapewniają mechanizm rozszerzonej obsługi wejścia/wyjścia. Javacardx.framework.event zawiera klasy oraz interfejsy obsługujące różne źródła zdarzeń, natomiast javacardx.framework.nio definiuje zoptymalizowane środki dostępu do danych (dane surowe lub dane ustrukturyzowane) w różnych lokalizacjach pamięci (pamięć wewnętrzna, zewnętrzna lub zmapowana).

Poza wyżej wymienionymi usprawnieniami, Java Card w wersji 3.1.0 wprowadza także kilka innych nowości [31]. Jedną z nich jest implementacja chińskich algorytmów SM2, SM3 oraz SM4. Algorytm SM2 jest algorytmem podpisu elektronicznego z zastosowaniem krzywych eliptycznych. Poza tym jest on odpowiedzialny za szyfrowanie klucza publicznego oraz wymianę kluczy. Obsługa tego algorytmu wymagała rozszerzenia już istniejących klas Signature, Cipher oraz KeyAgreement. SM3 jest natomiast algorytmem haszującym. Na jego potrzeby rozszerzona musiała zostać klasa MessageDigest. Ostatni z algorytmów - SM4 to z kolei algorytm szyfrowania blokowego. Aby możliwa była jego implementacja konieczne było

rozszerzenie klasy Cipher, a także dodanie nowego typu klucza SM4 wraz z odpowiadającym mu interfejsem.

Oprócz tego do najnowszej wersji Java Card dodane zostały klasy do obsługi czasu systemowego - SysTime oraz TimeDuration. Dzięki ich obecności w pakiecie javacardx.framework.time, możliwe jest dokonywanie porównań i obliczeń arytmetycznych na znacznikach czasu (ang. timestamp) oraz konwersja czasu na różne jednostki. Ostatnią rzeczą, o której warto wspomnieć jest dodanie do Java Card 3.1.0 zbioru ustalonych parametrów dotyczących kryptografii krzywych eliptycznych. Dotychczas programista musiał ręcznie skonfigurować każdy obiekt klucza za pomocą domenowych parametrów krzywych. Predefiniowane parametry dodane w wersji 3.1.0 pozwalają na szybkie stworzenie i użycie klucza bez potrzeb ręcznej konfiguracji.

5. Projekt jELS w wersji 2

W rozdziale tym omówione zostaną wymagania funkcjonalne i pozafunkcjonalne dotyczące apletu ELS 2. Przedstawiona zostanie struktura stworzonego projektu, a także wykorzystane narzędzia deweloperskie. W podrozdziale 5.4 opisane zostaną przeprowadzone testy funkcjonalne powstałej aplikacji.

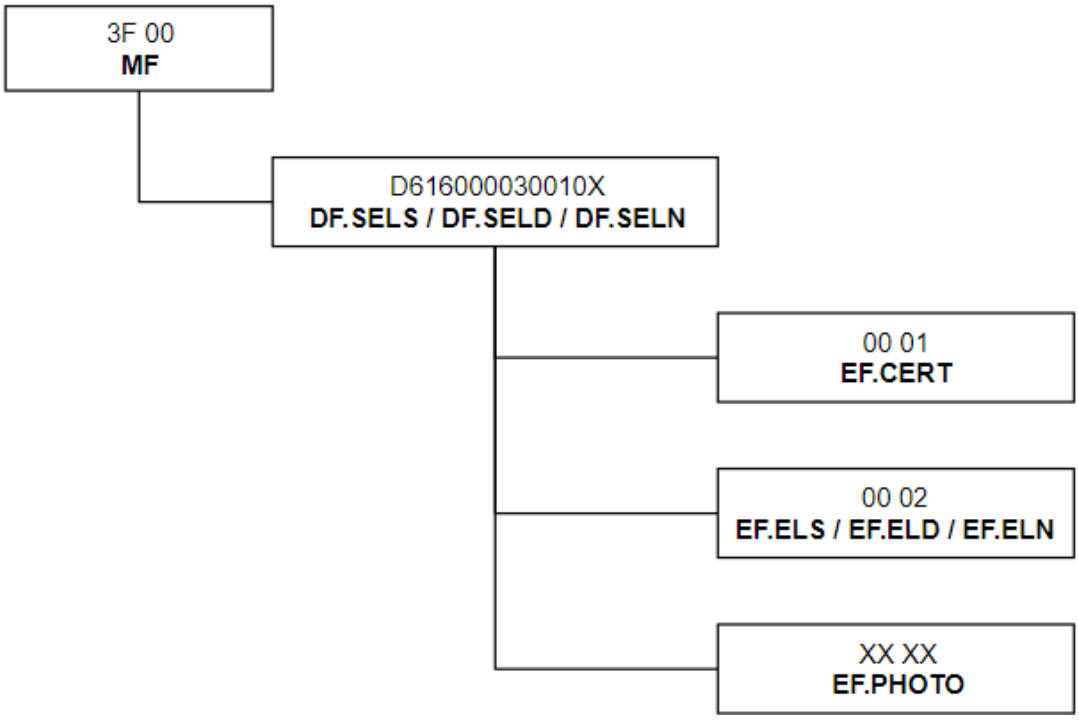
5.1 Wymagania funkcjonalne

Przez wymagania funkcjonalne rozumie się szereg funkcji, które musi oferować stworzony aplet, aby mógł on pełnić rolę Elektronicznej Legitymacji Studenckiej w wersji 2. Aplet będący przedmiotem tej pracy powstał na bazie dotychczas używanego apletu jELS w wersji 1, zgodnego z Rozporządzeniem Ministra Edukacji Narodowej i Sportu z dnia 18 lipca 2005 r. [17]. W związku z wejściem w życie nowego Rozporządzenia Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. [18], konieczne było wprowadzenie istotnych zmian w strukturze oraz w obsłudze plików znajdujących się na karcie.

5.1.1 Struktura plików jELS 2

W nowej wersji struktury jELS (rysunek 5.1) oprócz istniejących wcześniej plików DF.SELS, EF.CERT oraz EF.ELS dodany został również plik EF.PHOTO przechowujący fotografię cyfrową studenta w formacie JPEG. Plik ten posiada ID ustalane na etapie instalacji, które jest następnie zapisywane w dodatkowym polu w EF.ELS. Umieszczenie nowych pól w EF.ELS oraz rozszerzenie możliwości certyfikacji o pieczęć elektroniczną w EF.CERT sprawiło, że konieczne było zwiększenie maksymalnego rozmiaru tych plików na karcie. W celu ułatwienia migracji na nową wersję apletu, również w procesie instalacji udostępniona została możliwość wyboru między starą a nową wersją struktury plików.

Dodatkową dostępną opcją jest wybór rodzaju legitymacji, od którego zależy także AID aplikacji. W wersji 1. jELS dostępne były dwa typy: Elektroniczna Legitymacja Studencka (ELS) o AID równym D6160000300101 oraz Elektroniczna Legitymacja Doktorancka (ELD) o AID równym D6160000300102. Nowa wersja apletu jELS wprowadza także trzeci rodzaj, czyli Elektroniczną Legitymację Nauczyciela Akademickiego (ELNA) o AID równym D6160000300103.



Rysunek 5.1 Struktura plików jELS w wersji 2.

5.1.2 Polecenia APDU dla jELS

Zestaw obsługiwanych poleceń APDU dostępnych dla apletu Elektronicznej Legitymacji Studenckiej jest zgodny z normą ISO/IEC 7816-4 i składa się on jedynie z kilku opisanych w tym standardzie funkcji. Po pomyślnej instalacji apletu na karcie możliwe jest wykonanie jednego z następujących poleceń APDU: SELECT FILE, READ BINARY oraz UPDATE BINARY. Polecenie SELECT FILE pozwala na wybór jednego z plików znajdujących się na karcie. Wybrany plik staje się wówczas plikiem aktywnym, na którym można przeprowadzić operacje odczytu i zapisu. Poniżej przedstawiony został opis poleceń APDU wykorzystywanych do komunikacji z kartą Elektronicznej Legitymacji Studenckiej.

SELECT FILE (INS = 0xA4)

Polecenie to pozwala na wybór jednego z plików znajdujących się na karcie.

Parametry:

P1	P2	Opis
0x00	0x00	Wybór pliku MF, EF.CERT, EF.ELS lub EF.PHOTO. W polu danych znajduje się ID pliku.
0x02	0x00	

W odpowiedzi aplet wysyła strukturę FCI (ang. *File Control Information*) oznaczoną tagiem 6F. Zawarte są w niej następujące parametry:

Tag	Długość	Opis	Typ pliku
0x80	2	Liczba bajtów danych z wyłączeniem informacji o strukturze	Transparentny EF
0x81	2	Liczba bajtów danych wraz z informacjami o strukturze	Dowolny
0x82	1	Bajt deskryptora pliku	Dowolny
0x82	2	Bajt deskryptora pliku wraz z bajtem kodowania danych	Dowolny
0x82	3 lub 4	Bajt deskryptora pliku wraz z bajtem kodowania danych i maksymalną długością rekordu	EF ze strukturą rekordów
0x83	2	Identyfikator pliku	Dowolny
0x84	od 1 do 16	Nazwa DF	DF
0x85	zmienna	Informacje zastrzeżone	Dowolny
0x86	zmienna	Właściwości bezpieczeństwa	Dowolny
0x87	2	Identyfikator EF zawierającego rozszerzenie FCI	Dowolny

Komunikaty błędów:

SW1	SW2	Opis
0x69	0x84	Błędne dane.
0x6A	0x86	Błędne parametry P1 i P2.
0x6A	0x82	Nie znaleziono pliku o podanym ID.

READ BINARY (INS = 0xB0)

Polecenie to umożliwia odczyt danych z aktywnego pliku lub z pliku, którego ID podano jako parametr polecenia.

Parametry:

P1	P2	Opis
0x8X	0xXX	Odczyt z pliku EF, którego krótki identyfikator podany jest na bitach od 5 do 1 parametru P1. P2 oznacza offset.
0xYX dla Y < 8	0xXX	Odczyt z aktywnego pliku. P1 || P2 oznacza offset.

Dla Y > 8 wartość parametru P1 jest nieprawidłowa.

Jako odpowiedź aplet przesyła odpowiedni fragment danych z wybranego pliku.

Komunikaty błędów:

SW1	SW2	Opis
0x69	0x86	Niedozwolone polecenie (brak aktywnego pliku).
0x6B	0x00	Błędne parametry P1 i P2.
0x6A	0x82	Nie znaleziono pliku o podanym ID.
0x67	0x00	Podano błędną długość odczytu.
0x62	0x82	Osiągnięto koniec pliku.

UPDATE BINARY (INS = 0xD6)

Polecenie to pozwala na nadpisanie zawartości danych w pliku aktywnym lub w pliku, którego ID podano jako parametr polecenia.

Parametry:

P1	P2	Opis
0x8X	0xXX	Zapis do pliku EF, którego krótki identyfikator podany jest na bitach od 5 do 1 parametru P1. P2 oznacza offset.
0xYX dla Y < 8	0xXX	Zapis do aktywnego pliku. P1 || P2 oznacza offset.

Dla Y > 8 wartość parametru P1 jest nieprawidłowa.

Jako odpowiedź na polecenie karta wysyła jedynie bajty statusu.

Komunikaty odpowiedzi:

SW1	SW2	Opis
0x69	0x86	Niedozwolone polecenie (brak aktywnego pliku).
0x6B	0x00	Błędne parametry P1 i P2.
0x6A	0x82	Nie znaleziono pliku o podanym ID.
0x69	0x82	Niewystarczający stan bezpieczeństwa.
0x90	0x00	Poprawne wykonanie zapisu.

5.2 Wymagania pozafunkcjonalne

Istotne jest, aby nowa wersja apletu jELS spełniała poza wymaganiami funkcjonalnymi również określone wymagania pozafunkcjonalne. Wymagania te są ściśle związane z jakością działania aplikacji, a także m.in. z jej wydajnością, niezawodnością i bezpieczeństwem [33]. Poniżej przedstawione zostały wymagania pozafunkcjonalne określone na etapie analizy projektu.

Opis	Aplet powinien zajmować możliwie jak najmniej miejsca na karcie elektronicznej.
Kategoria	Wydajność
Priorytet	Średni

Opis	Aplet po otrzymaniu danych musi przechowywać je w niezmienionej formie. Odczytane dane nie są modyfikowane.
Kategoria	Funkcjonalne dopasowanie
Priorytet	Wysoki

Opis	Aplet musi umożliwiać współistnienie na jednej karcie z innymi apletami i nie może ograniczać ich funkcjonalności.
Kategoria	Kompatybilność
Priorytet	Wysoki

Opis	Aplet można zainstalować na kartach z systemem Java Card Classic 3.0.5 oraz GlobalPlatform 1.6.
Kategoria	Kompatybilność
Priorytet	Wysoki

Opis	Odpowiedzi apletu na polecenia powinny w jasny sposób informować użytkownika o statusie wykonania operacji.
Kategoria	Użyteczność
Priorytet	Wysoki

Opis	Po wystąpieniu awarii można odinstalować i zainstalować aplet na nowo.
Kategoria	Niezawodność
Priorytet	Średni

Opis	Aplet powinien blokować wykonywanie nieautoryzowanych operacji.
Kategoria	Bezpieczeństwo
Priorytet	Wysoki

Opis	Aplet powinien być przygotowany wraz z dodatkowymi skryptami umożliwiającymi jego łatwe testowanie.
Kategoria	Łatwość utrzymania
Priorytet	Niski

5.3 Implementacja rozwiązania

W rozdziale tym przedstawiono wykorzystane środowisko deweloperskie, opisano strukturę projektu oraz sposób instalacji i implementacji apletu jELS w wersji 2. Na koniec omówiono zastosowane w projekcie mechanizmy bezpieczeństwa.

5.3.1 Opis środowiska deweloperskiego

Projekt nowej wersji jELS został zrealizowany na laptopie z 64-bitowym systemem operacyjnym Windows 8.1. W celu przetestowania apletu w środowisku rzeczywistym przygotowane zostały 2 karty JCOP 4 z MIFARE oraz czytnik kart Techly USB 2.0 z chipsetem Realtek RTS5169.

Do stworzenia kodu aplikacji, konfiguracji oraz kompilacji projektu wykorzystane zostało środowisko programistyczne Eclipse IDE w wersji 4.11.0. Środowisko to posiada dedykowany interfejs przeznaczony do łatwej obsługi projektów Java Card. Zbudowanie apletu w Eclipse IDE wymagało dołączenia do niego następujących bibliotek: Java Card Development Kit 3.0.5 (pakiet ten można pobrać ze strony Oracle) oraz GlobalPlatform w wersji 1.6 (pakiet dostępny na oficjalnej stronie globalplatform.org).

5.3.2 Struktura projektu

Struktura utworzonego projektu Java Card została przedstawiona na rysunku 5.2. W katalogu *src* znajdują się dwie klasy: jELS oraz TransparentFile. Klasa jELS rozszerza klasę Applet z Java Card i zawiera funkcje niezbędne do obsługi Elektronicznej Legitymacji Studenckiej. Z kolei TransparentFile jest klasą pomocniczą, w której zawarte są funkcje wspierające operacje odczytu i zapisu do plików znajdujących się na karcie. W folderze *Referenced Libraries* umieszczone zostały biblioteki wymienione w podrozdziale 5.3.1. Katalog *apdu_scripts* zawiera automatycznie generowane na etapie kompilacji skrypty, które weryfikują poprawność pliku .cap, a także testują instalację i wybór apletu. W kolejnym folderze *configurations* znajdują się pliki konfiguracyjne: jELS.conf oraz jELS.scriptgen. Plik jELS.conf definiuje parametry wykorzystywane na etapie budowania projektu (rysunek 5.3):

`-i` – włącza obsługę 32-bitowego typu int;

`-classdir` – określa folder, w którym znajdują się skompilowane pliki typu .class;

`-exportpath` – określa ścieżkę, w której umieszczone zostały pliki .exp (m.in. GlobalPlatform);

`-applet` – ustawia domyślny AID apletu oraz określa nazwę klasy, która jest jego definicją;

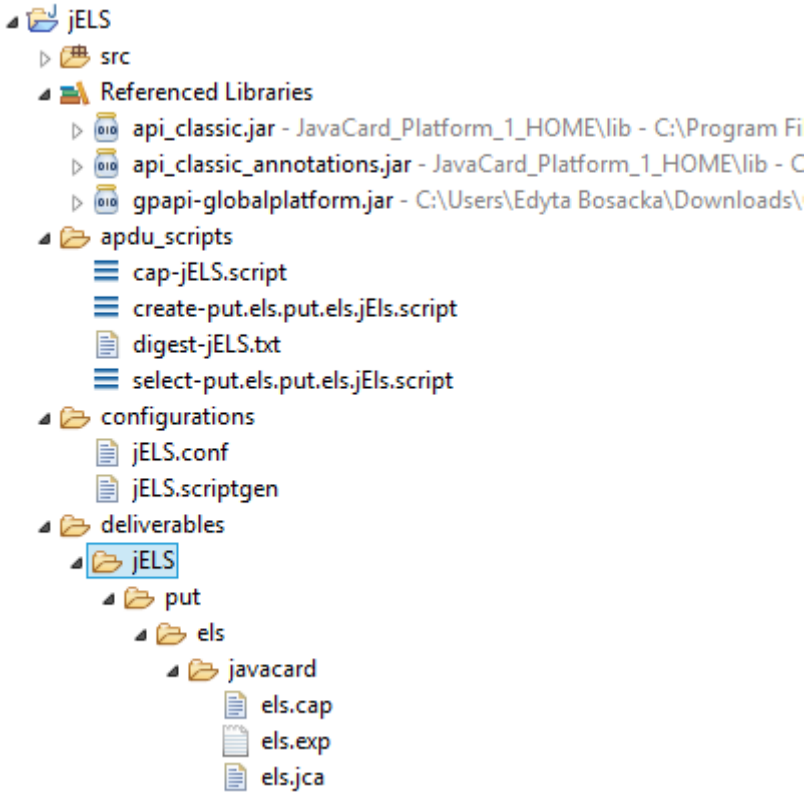
`-out` – określa typy plików, które zostaną wygenerowane przez konwerter (do wyboru pliki .cap, .exp i .jca);

`-d` – wybiera folder do zapisu wygenerowanych plików;

`-debug` – włączenie tej opcji powoduje wygenerowanie opcjonalnego elementu debug.cap w pliku .cap apletu;

`-target` – określa wersję platformy Java Card, dla której wygenerowany zostanie plik .cap;

`-package-name, package-aid, major-version, minor-version` – definiuje nazwę pakietu, jego AID, a także wersję zdefiniowaną przez użytkownika.



Rysunek 5.2 Struktura projektu jELS 2 w Eclipse IDE.

jELS.conf

```
-i
-classdir ..\bin
-exportpath C:\api_export\
-applet 0xD6:0x16:0x00:0x00:0x30:0x01:0x01 put.els.jELS
-out EXP JCA CAP
-d ..\deliverables\jELS
-debug
-target 3.0.5
put.els
0xD6:0x16:0x00:0x00:0x30:0x01 2.0
```

Rysunek 5.3 Zawartość pliku konfiguracyjnego jELS.conf.

Ostatni folder *deliverables* przechowuje wygenerowane przez konwerter pliki els.cap, els.exp i els.jca. els.cap jest plikiem zawierającym wcześniej skompilowane i skonwertowane klasy apletu, który wgrywany jest na kartę elektroniczną. els.exp zawiera pola i metody, które mogą być importowane przez inne aplety. Z kolei els.jca jest reprezentacją tekstową zawartości pliku .cap.

5.3.3 Instalacja apletu jELS 2.0

Jednym z wymagań funkcjonalnych jELS w wersji 2 jest zapewnienie możliwości wyboru pomiędzy starą a nową wersją struktury plików. Zdecydowano, że wybór ten będzie dokonywany na etapie instalacji apletu poprzez wysłanie polecenia APDU z odpowiednimi parametrami. W celu zainstalowania jELS z nową strukturą plików należy podać następujące parametry:

Le Le_ver Ver Le_id Id,

gdzie:

- Le oznacza długość łańcucha parametrów (w bajtach),
- Le_ver jest długością parametru oznaczającego wersję struktury plików (wartość ta powinna wynosić 01),
- Ver oznacza wybraną wersję struktury plików jELS (01 dla wersji 1, 02 dla wersji 2),
- Le_id jest długością parametru określającego ID, z jakim zostanie utworzony plik EF.PHOTO (wartość ta powinna wynosić 02),
- Id oznacza wybrany identyfikator pliku EF.PHOTO.

Poniżej przedstawiony został przykładowy łańcuch parametrów, który pozwoli na zainstalowanie jELS z nową wersją struktury plików oraz na utworzenie pliku EF.PHOTO z ID równym 0004:

05 01 02 02 00 04

5.3.4 Implementacja jELS 2.0

Projekt jELS 2.0 w dużej mierze opiera się na implementacji apletu jELS w wersji 1. Jednak ze względu na nowe wymagania zawarte w Rozporządzeniu Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. konieczne było dokonanie odpowiednich zmian w kodzie apletu. Pierwszą przeprowadzoną modyfikacją było dodanie w konstruktorze apletu obsługi parametrów instalacji opisanych w podrozdziale 5.3.3. Fragment odpowiedzialny za tę funkcjonalność przedstawiono na listingu 5.1.

```
protected jELS(byte[] bArray, short bOffset, byte bLength) throws ISOException {

    byte iLen = bArray[bOffset]; // AID length
    bOffset = (short) (bOffset+iLen+1);
    byte cLen = bArray[bOffset]; // info length
    bOffset = (short) (bOffset+cLen+2);
    //applet parameters
    byte versionLength = bArray[bOffset]; // "ELS structure version" parameter length
    bOffset = (short)(bOffset+1);

    version = bArray[bOffset];

    //create ef.cert
    EF_CERT = new TransparentFile((short)0x0001, (short)0x1000);

    //create ef.els
    EF_ELS = new TransparentFile((short)0x0002, (short)0x0C00);

    if(version == VERSION2) // for version 2.0 of file structure
    {
        bOffset = (short)(bOffset+versionLength+1);
        efPhotoId = Util.makeShort(bArray[bOffset], bArray[(short)(bOffset + 1)]);

        //create ef.photo
        EF_PHOTO = new TransparentFile(efPhotoId,(short) 0x7F00);
    }
```

Listing 5.1 Fragment kodu odpowiedzialny za obsługę parametrów instalacji.

Kolejną ważną modyfikacją dodaną do nowej wersji jELS była możliwość wyboru pliku EF.PHOTO za pomocą polecenia SELECT FILE (tylko w przypadku, gdy plik ten został wcześniej utworzony). Powstały fragment kodu pokazany został na listingu 5.2.

```
default:
    if(version == VERSION2 && id == efPhotoId) //selection of EF.PHOTO
    {
        currentFile = EF_PHOTO;
        temp[4] = (byte) 0x7F;
        temp[11] = buffer[ISO7816.OFFSET_CDATA];
        temp[12] = buffer[ISO7816.OFFSET_CDATA + 1];
        temp[14] = buffer[ISO7816.OFFSET_CDATA];
        temp[15] = buffer[ISO7816.OFFSET_CDATA + 1];
        apdu.setOutgoing();
        apdu.setOutgoingLength((short) 16);
        apdu.sendBytesLong(temp, (short) 0, (short) 16);
        break;
    }
    currentFile = null;
    ISOException.throwIt(ISO7816.SW_FILE_NOT_FOUND);
```

Listing 5.2 Fragment kodu umożliwiający wybór pliku EF.PHOTO.

Podobne zmiany zastosowano w przypadku obsługi poleceń READ BINARY i UPDATE BINARY, w których jednym z parametrów jest identyfikator pliku (listing 5.3).

```
default:
    if(version == 0x02 && (p1 & 0x1F) == (efPhotoId & 0x1F))
    {
        currentFile = EF_PHOTO; //selection of EF.PHOTO
        break;
    }
    else
    {
        ISOException.throwIt(ISO7816.SW_FILE_NOT_FOUND);
    }
```

Listing 5.3 Fragment kodu odpowiedzialny za wybór pliku EF.PHOTO podczas operacji zapisu i odczytu.

5.3.5 Mechanizmy bezpieczeństwa

Przechowywane w aplecie jELS pliki zawierają informacje, które nie mogą być zmieniane przez osobę do tego nieupoważnioną, dlatego istotne jest zapewnienie bezpieczeństwa ich przetwarzania. W tym celu z pomocą przychodzą mechanizmy zawarte w specyfikacji Global Platform. Aby zapewnić ochronę przesyłanych danych, zdecydowano się na wykorzystanie mechanizmu *Secure Channel*. Polega on na utworzeniu bezpiecznego połączenia (kanału) pomiędzy kartą a czytnikiem na czas trwania jednej sesji. Jest to możliwe tylko wówczas, gdy użytkownik zna sekretny klucz znajdujący się w *Security Domain* karty. Po pomyślnej weryfikacji tego klucza,

następuje zainicjalizowanie bezpiecznego kanału, który pozwala na ochronę kryptograficzną przesyłanych informacji. Utworzenie bezpiecznego kanału jest warunkiem koniecznym dla przeprowadzenia operacji zapisu w nowej wersji apletu jELS.

5.4 Testy apletu jELS realizującego strukturę w wersji 2

Ostatni etap tworzenia oprogramowania polega na przetestowaniu aplikacji pod kątem jej funkcjonalności i niezawodności. W celu przeprowadzenia testów apletu jELS w wersji 2 przygotowane zostały dwie karty firmy NXP Semiconductors: JCOP 3 P60 oraz JCOP 4 P71. W tabeli 5.1 przedstawione zostały wybrane parametry tych kart:

Karta testowa	JCOP 3 P60	JCOP 4 P71
Wersja Java Card	3.0.4 Classic	3.0.5 Classic
Wersja GlobalPlatform	2.2.1	2.3
Dostępna pamięć FLASH	do 108 kB	do 180 kB
Protokół bezpiecznego kanału (SCP)	SCP 02	SCP 01, 02 oraz 03

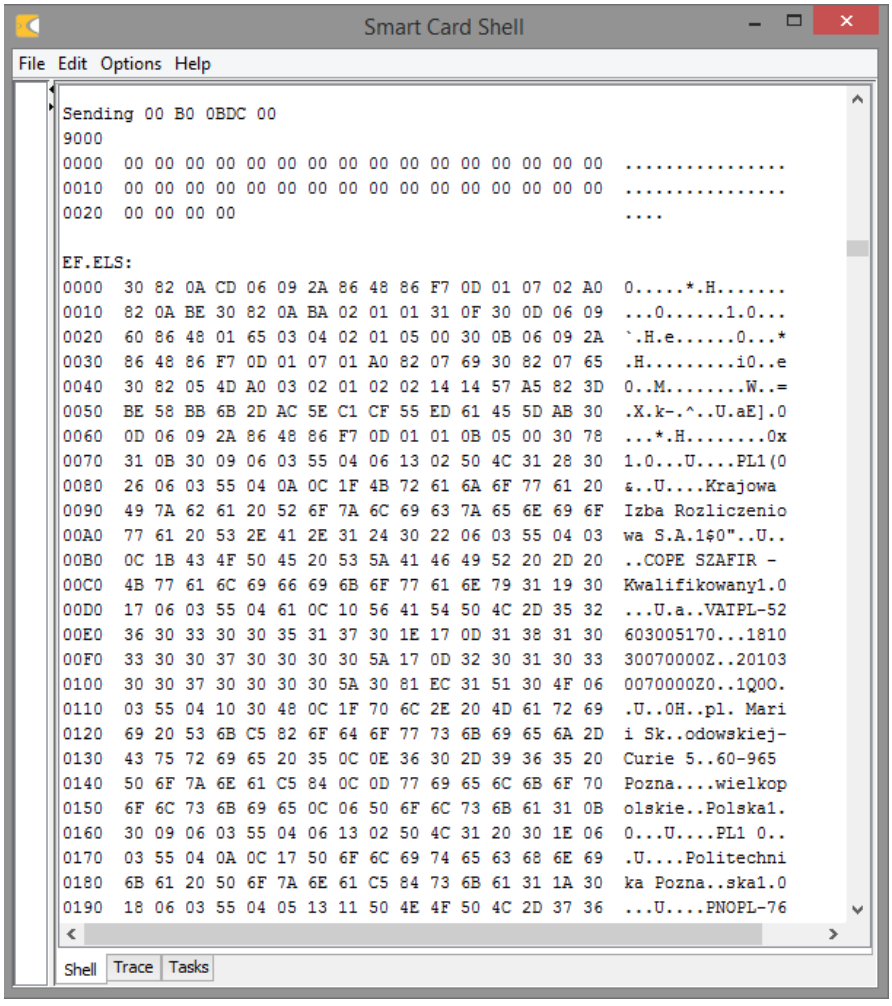
Tabela 5.1 Parametry kart testowych

Do komunikacji z kartą wykorzystany został program Smart Card Shell korzystający z interpretera i kompilatora JavaScript oraz pozwalający na wykonywanie skryptów napisanych w tym języku. Skrypty testowe dla jELS 2.0 powstały poprzez zmodyfikowanie istniejących już skryptów weryfikujących poprawność działania wersji 1 apletu. W dalszej części pracy opisane zostały przypadki testowe sprawdzające poprawność nowego rozwiązania. Testy te zakończyły się powodzeniem, o ile w ich szczegółowym opisie nie zaznaczono inaczej.

Pierwszy test polegał na zainstalowaniu apletu na karcie, utworzeniu dedykowanych plików oraz wgraniu do nich przykładowych danych. Przeprowadzono go w trzech wariantach: dla ELS (z nową wersją struktury plików), ELD oraz ELNA. W ramach testu dla karty P71 udało się oszacować maksymalny rozmiar przechowywanego na niej pliku EF.PHOTO, który wynosi 0x7F00. Niestety wgranie jELS 2.0 na kartę P60 nie powiodło się. Przyczyną tego może być zarówno niezgodność karty z wykorzystywaną w aplecie wersją Java Card oraz wersją GlobalPlatform, a także fakt, iż karta P60 posiada znacznie mniejszą ilość dostępnej pamięci niż karta

P71. Z tego powodu dalsze testy przeprowadzane były wyłącznie na karcie JCOP 4 P71.

Kolejny test miał na celu sprawdzenie poprawności wgrania apletu oraz plików na kartę poprzez ich wybór za pomocą polecenia SELECT. Rozbudowaną wersją tego testu jest test trzeci, polegający na przeskanowaniu struktury plików apletu i weryfikacji odpowiedzi karty na próbę wyboru pliku o dowolnym identyfikatorze. Operacja ta powinna zakończyć się sukcesem jedynie dla istniejących plików, w przeciwnym wypadku zwrócony zostanie komunikat o błędzie: 0x6A82. W następnym teście sprawdzono poprawność odczytu plików zapisanych na karcie. W tym celu skrypt najpierw wybiera kolejne pliki, po czym wysyła w pętli polecenie READ BINARY. W odpowiedzi aplet przekazuje odpowiedni fragment wybranego pliku (rysunek 5.4).



Rysunek 5.4 Fragment wyników skryptu odczytującego zawartość plików.

Test piąty weryfikował z kolei poprawność operacji zapisu i odczytu danych z plików znajdujących się na karcie. Sprawdzenie to polegało na zapisaniu losowego

fragmentu pliku za pomocą polecenia UPDATE BINARY, a następnie odczytaniu go poleceniem READ BINARY. Test ten miał również pełnić rolę testu obciążeniowego poprzez wywołanie wyżej wymienionych poleceń w pętli. W czasie testowania zaobserwowano, że karta może poprawnie obsłużyć około tysiąca par operacji zapis-odczyt, po czym przestaje ona odpowiadać. Problem ten próbowano rozwiązać na kilka sposobów m.in. weryfikując poprawność zarządzania pamięcią karty, a także dodając timer, którego zadaniem było zresetowanie karty po dłuższym okresie bezczynności. Modyfikacje te nie rozwiązały jednak opisanego problemu. Aczkolwiek uznano, że nie jest on krytyczny, ponieważ karta po odłączeniu zasilania i ponownym podłączeniu była w dalszym ciągu sprawna. Dodatkowo warto mieć na uwadze fakt, iż w rzeczywistym przypadku użytkowania, wykonanie większej ilości operacji podczas jednej sesji jest mało prawdopodobne. W kolejnym teście sprawdzono możliwość aktualizacji wybranego fragmentu pliku. Stworzony w tym celu skrypt wybierał jeden z plików znajdujących się na karcie (do testowania wybrano plik EF.CERT), a następnie modyfikował część zawartych w nim danych. Ostatni etap testu polegał na porównaniu początkowej i końcowej zawartości pliku. Jeśli różniły się one od siebie, znaczyło to, że aktualizacja pliku została pomyślnie wykonana. Oprócz zapewnienia prawidłowego działania apletu ELS w wersji 2, istotne jest, aby spełniał on również podstawowe kwestie związane z bezpieczeństwem. Z tego względu w ostatniej fazie testowania sprawdzono możliwość obsługi wielu zestawów kluczy przez kartę. Napisany skrypt uwierzytelniał klucz apletu przy pomocy kluczy znajdujących się na karcie. W przypadku ich niezgodności wyświetlany był komunikat „Security condition not satisfied” (kod błędu: 6982).

6. Wnioski

Celem pracy było stworzenie apletu Elektronicznej Legitymacji Studenckiej w wersji 2 zgodnej z Rozporządzeniem Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. [18]. Cel ten został w pełni zrealizowany. W części teoretycznej pracy porównano starą i nową wersję ELS oraz omówiono mechanizmy ich bezpieczeństwa. Dokonano także przeglądu dostępnych wersji Java Card w celu wyboru takiej, która pozwoliłaby na najefektywniejsze stworzenie nowoczesnego apletu korzystającego z funkcjonalności zawartych w 1. wersji jELS. Efektem praktycznym pracy jest zaimplementowany aplet jELS 2.0, umożliwiający przechowywanie pliku ze zdjęciem studenta oraz wykonywanie na nim operacji zapisu i odczytu. Aplet ten może być zainstalowany zarówno jako Elektroniczna Legitymacja Studencka (ELS), jak i Elektroniczna Legitymacja Doktorancka (ELD) oraz Elektroniczna Legitymacja Nauczyciela Akademickiego (ELNA). W ramach pracy przygotowane zostały również testy funkcjonalne sprawdzające odpowiedzi karty na dostępne polecenia APDU. Testy te zostały przeprowadzone na specjalnie przygotowanych przez uczelnię kartach. Kolejnym etapem projektu będzie integracja nowej wersji apletu jELS z istniejącymi systemami Politechniki Poznańskiej. W momencie ukończenia tej pracy faza ta jeszcze się nie rozpoczęła, jednak wdrożenie ELS w wersji 2 planowane jest w niedalekiej przyszłości.

Chociaż prace nad apletem jELS 2.0 zostały pomyślnie przeprowadzone, w dalszym ciągu istnieje możliwość potencjalnego rozwoju stworzonego rozwiązania. Jednym ze sposobów unowocześnienia apletu byłaby jego migracja z Java Card 3.0.5 Classic na Java Card 3.0.5 Connected, która oferuje szereg nowych funkcjonalności m.in. wzbogacona została o nową 32-bitową maszynę wirtualną, automatyczne „odśmiecanie pamięci” oraz obsługę wielowątkowości. Innym sposobem byłoby dodanie interfejsu *Shareable* w celu współdzielenia wybranych funkcjonalności z innymi apletami na karcie. Udostępnienie im całości bądź części systemu plików pozwoliłoby na redukcję zajętości pamięci karty. Z drugiej strony istnieje też możliwość, że w przyszłości blankiet Elektronicznej Legitymacji Studenckiej zostanie całkowicie zastąpiony przez aplikację mLegitymacja dostępną na urządzeniach mobilnych.

Literatura

[1] Laxmi Ashrit, *What is Smart card – How it Works, Specifications, Types and Applications*, dostęp na dzień 05.05.2020 r.
https://electricalfundablog.com/smart-card-works-specifications-types/

[2] S.A.M Rizvi, Halima S. Rizvi, Zaid Al.-Baghdadi, *Smart Cards: The Future Gate*, October 2010.
http://www.iaeng.org/publication/WCECS2010/WCECS2010_pp81-86.pdf

[3] ISO/IEC 7810:2019 *Identification cards – Physical characteristics*, December 2019.

[4] *Elements of Smart Card Architecture*, dostęp na dzień 13.05.2020 r.
https://people.cs.uchicago.edu/~dinoj/smartcard/arch-1.html

[5] Zhiqun Chen, *Java Card™ Technology for Smart Cards*, Addison-Wesley Professional, 2000.

[6] *Types of Smart Card*, dostęp na dzień 13.05.2020 r.
http://www.smartcardbasics.com/smart-card-types.html

[7] *How to program Java Card 3.0 platforms?*, Samia Bouzefrane, 2019.
https://cedric.cnam.fr/~bouzefra/cours/JC_3.pdf

[8] *Smart Card Primer*, dostęp na dzień 18.05.2020 r.
https://www.securetechalliance.org/smart-cards-intro-primer/

[9] Tarun Agarwal, *How does the Smart Card Works?*, dostęp na dzień 30.05.2020 r.
https://www.elprocus.com/working-of-smart-card/

[10] Piotr Nazimek, *Inżynieria programowania kart inteligentnych*, Warszawa 2005.

[11] *ISO/IEC 7816*, dostęp na dzień 04.08.2020 r.
https://en.wikipedia.org/wiki/ISO/IEC_7816

[12] *ISO 7816 – Smart Card Standards Overview*, dostęp na dzień 04.08.2020 r.
http://www.smartcardsupply.com/Content/Cards/7816standard.htm

[13] *Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange.* ISO/IEC 7816-4. Edycja 3., 15.04.2013 r.

[14] *Identification cards – Integrated circuit cards – Part 9: Commands for card management.* ISO/IEC 7816-9. Edycja 3., 12.2007 r..

[15] M. Gosławski, *Electronic Student Identity Card Management System at the Poznan University of Technology*, 2011.

[16] Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 2 listopada 2006 r. w sprawie dokumentacji przebiegu studiów.
http://prawo.sejm.gov.pl/isap.nsf/download.xsp/WDU20062241634/O/D20061634.pdf
[17] Rozporządzenie Ministra Edukacji Narodowej i Sportu z dnia 18 lipca 2005 r. w sprawie dokumentacji przebiegu studiów.
http://prawo.sejm.gov.pl/isap.nsf/download.xsp/WDU20051491233/O/D20051233.pdf
[18] Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 16 kwietnia 2019 r. zmieniające rozporządzenie w sprawie studiów.
http://prawo.sejm.gov.pl/isap.nsf/download.xsp/WDU20190000787/O/D20190787.pdf
[19] Projekt rozporządzenia Ministra Nauki i Szkolnictwa Wyższego z dnia 21 grudnia 2018 r. (nr. DLP.ZLS.1201.40.2018.AK) zmieniającego rozporządzenie w sprawie studiów.
https://www.krasp.org.pl/resources/upload/dokumenty/dokumenty_opinie_16-20/Kkszta%C5%82cenia/projekt_rozp_zm_rozporzadzenie_studia.pdf
[20] Regulamin Usługi mLegitymacja studencka w Aplikacji mObywatel – dostępne funkcje, ochrona danych osobowych, postanowienia licencyjne, dostęp na dzień 24.08.2020 r.
https://www.mobywatel.gov.pl/mobywatel.android.mlegitymacjastudencka.regulamin.2.1 .0.pdf
[21] Dokumentacja eksploatacyjna dla pracownika szkoły i uczelni, wersja 1.14, Ministerstwo Cyfryzacji.
[22] M. Gosławski, *Elektroniczna Legitymacja Studencka*, II Krajowa Konferencja Użytkowników Systemów Elektronicznej Legitymacji Studenckiej z warsztatami, 10 czerwca 2010 r.
[23] Wymagania bezpiecznego użytkowania systemu, Ministerstwo Cyfryzacji.
[24] Informacje o publicznej aplikacji mobilnej, Ministerstwo Cyfryzacji.
[25] *Virtual Machine Specification, Java Card 3 Platform, Version 3.0.5, Classic Edition.* Oracle, 2015.
[26] *Runtime Environment Specification, Java Card 3 Platform, Version 3.0.5, Classic Edition.* Oracle, 2015.
[27] *Application Programming Interface, Java Card 3 Platform, Version 3.0.5, Classic Edition.* Oracle, 2015.
https://docs.oracle.com/javacard/3.0.5/api/index.html

[28] *GlobalPlatform Technology, Card Specification, Version 2.3.1.*, GlobalPlatform, 2018.
https://globalplatform.org/wp-content/uploads/2018/05/GPC_CardSpecification_v2.3.1_PublicRelease_CC.pdf
[29] Release Notes. *Specifications for the Java Card 3 Platform, Version 3.0.5, Classic Edition.* Oracle, 2017
[30] White Paper. *The Java Card™ 3 Platform.* Sun Microsystems, 2018.
[31] Release Notes. *Java Card Platform, Version 3.1.0*. Oracle, 2019.
[32] *What's new in Java Card 3.1*, Christian Hujer, 2019.
https://nelkinda.com/blog/whats-new-in-javacard-31/#d11e195
[33] Wymagania pozafunkcjonalne – projektowanie interfejsu użytkownika, dostęp na dzień 06.09.2020 r.
http://www.cs.put.poznan.pl/jrojek/files/io1/requirements/NFR_opis.pdf