INSTYTUT INFORMATYKI
WYDZIAŁ INFORMATYKI
POLITECHNIKA POZNAŃSKA

MAGISTERSKA PRACA DYPLOMOWA

PROJEKT I IMPLEMENTACJA UNIWERSALNEJ APLIKACJI DO OBSŁUGI KART ELEKTRONICZNYCH

inż. Mateusz LESZNER

Promotor:

dr inż. Marek MIKA

Opiekun:

mgr inż. Marek GOSŁAWSKI

Poznań 2012

SPIS TREŚCI

1. WSTĘP 7

CEL PRACY 7

ZAKRES PRACY 7

2. KARTY ELEKTRONICZNE 9

2.1. HISTORIA 9

2.2. BUDOWA 10

2.3. KOMUNIKACJA 13

2.4. KLASYFIKACJA KART 17

2.5. PODSUMOWANIE 21

3. ZAJĘCIA LABORATORYJNE 22

3.1. PRZEDSTAWIENIE SCENARIUSZY ZAJĘĆ ORAZ WARUNKÓW ZALICZENIA PRZEDMIOTU 28

3.2. ZAPOZNANIE Z KARTAMI 28

3.3. ELEKTRONICZNA LEGITYMACJA STUDENTA 29

3.4. INTERFEJS BEZSTYKOWY – PAMIĘĆ MIFARE 31

3.5. KARTY SIM 32

3.6. KARTY PŁATNICZE EMV 33

3.7. MIĘDZYUCZELNIANE CENTRUM PERSONALIZACJI LEGITYMACJI STUDENCKICH 34

3.8. KARTY BASICCARD 35

3.9. GLOBAL PLATFORM I JAVACARD 36

3.10. KARTY .NET 37

3.11. SKRYPTY W SMART CARD SHELL 38

3.12. PROJEKT 39

4. PROJEKT APLIKACJI 40

4.1. WYMAGANIA FUNKCJONALNE 40

4.2. WYMAGANIA POZAFUNKCJONALNE 41

4.3. ARCHITEKTURA 42

4.4. GŁÓWNA APLIKACJA 43

4.5. WTYCZKI 44

5. IMPLEMENTACJA 45

5.1. IMPLEMENTACJA GŁÓWNEJ APLIKACJI 45

5.2. IMPLEMENTACJA WTYCZEK 47

6. PODSUMOWANIE 49

WSKAZÓWKI DALSZEGO ROZWOJU 49

LITERATURA 50

DODATEK A – LISTA UŻYWANYCH SKRÓTÓW 52

DODATEK B – SPIS RYSUNKÓW I TABEL 53

SPIS RYSUNKÓW 53

SPIS TABEL 53

DODATEK C – ZAWARTOŚĆ DVD 54

1. WSTĘP

Karty elektroniczne są bardzo popularne w życiu codziennym. Każdego dnia wielu ludzi używa kart elektronicznych jako środka płatniczego. Służą także jako identyfikator w systemach kontroli dostępu. Zastosowania kart elektronicznych są olbrzymie. Tak duża popularność powoduje, że słuszne jest kształcenie studentów w temacie programowania kart elektronicznych. Niniejsza praca przedstawia propozycję usprawnienia pracy laboratorium programowania kart elektronicznych poprzez dostarczenie uniwersalnego narzędzia obsługującego różnego rodzaju karty elektroniczne.

Cel pracy

Istnieje wiele rodzajów kart elektronicznych, które różnią się interfejsami dostępu, protokołami komunikacyjnymi a także sposobem obsługi. W związku z powyższym trudno jest przygotować jedną aplikację, która będzie potrafiła obsłużyć wszystkie karty. Tego rodzaju oprogramowanie musi być ciągle rozwijane, ponieważ pojawiają się nowe karty elektroniczne, różniące się sposobem obsługi od wcześniejszych kart. Dotychczas podczas zajęć laboratoryjnych z przedmiotu programowanie kart elektronicznych były używane różne aplikacje do obsługi odpowiednich kart. Duża liczba aplikacji utrudnia zrozumienie zasad programowania kart elektronicznych, ponieważ najpierw należy nauczyć się obsługi danego programu.

Celem niniejszej pracy jest przedstawienie scenariuszy zajęć laboratoryjnych oraz przygotowanie uniwersalnej aplikacji obsługującej wiele rodzajów kart elektronicznych, umożliwiającej implementację obsługi nie przewidzianych wcześniej kart. Ważne jest, aby rozbudowa aplikacji była możliwa i prosta, a także, by nowe funkcje nie wpływały destrukcyjnie na pozostałą część oprogramowania. Efektem niniejszej pracy jest oprogramowanie udostępnione na licencji GNU GPL umożliwiającej bezproblemowy rozwój i publikację aplikacji.

Motywacją do zdefiniowania zakresu pracy była chęć zastąpienia dużej liczby różnych programów wykorzystywanych w trakcie prowadzenia zajęć jednym, uniwersalnym narzędziem, możliwym do dalszej rozbudowy w miarę rozwoju kart elektronicznych przez kolejne grupy laboratoryjne lub prace dyplomowe.

Zakres pracy

Rozdział 2 zawiera podstawy teoretyczne wymagane do zrozumienia tematu kart elektronicznych. Przedstawiona została historia kart elektronicznych, które były poprzedzone kartami papierowymi i blaszanymi. W dalszej części są opisane najpopularniejsze formaty kart elektronicznych. Na szczególną uwagę zasługuje podrozdział zatytułowany „Komunikacja”, w którym zostały opisane protokoły komunikacyjne oraz zestaw standardowych poleceń, które można przesłać do karty. W podrozdziale „Klasyfikacja” przedstawiono podział kart ze względu na kilka podstawowych kryteriów.

W rozdziale 3 zostały opisane programy używane dotychczas podczas zajęć laboratorium programowania kart elektronicznych. Niektóre z nich są trudno dostępne – nie ma możliwości ich pobrania ze strony internetowej producenta. Kolejne podrozdziały przedstawiają przykładowe scenariusze zajęć laboratoryjnych. Zawierają one krótki opis zajęć, zadania, które studenci powinni wykonać w ramach danych zajęć oraz przykładowe pytania sprawdzające wiedzę.

Rozdział 4 przedstawia projekt oprogramowania nazwanego „SmartCard Suite" w wersji 2.0, którego prototyp został przygotowany w ramach projektu realizowanego na zajęciach laboratoryjnych przedmiotu programowanie kart elektronicznych. Na podstawie zdobytego doświadczenia zaprojektowano nową, stabilną wersję, łatwą w rozbudowie.

W rozdziale 5 przedstawiono sposób implementacji oprogramowania, użyte biblioteki zewnętrzne oraz opisano sposób komunikacji pomiędzy główną aplikacją i wtyczkami..

Rozdział 6 zawiera wskazówki dotyczące dalszego rozwoju oprogramowania.

2. KARTY ELEKTRONICZNE

W niniejszym rozdziale przedstawiono teoretyczne wiadomości na temat kart elektronicznych, ich historię, formaty oraz budowę. W dalszej części przedstawiono klasyfikacje kart ze względu na różne kryteria.

2.1. Historia

Historia kart elektronicznych jest ściśle związana z historią kart płatniczych, o których pierwsza wzmianka pojawiła się już w roku 1887 w powieści Edwarda Bellamy „W roku 2000.” (org. *Looking backward*) [2], [4]. Pierwsze karty płatnicze pojawiły się w latach 20-tych XX wieku - były to karty (nie elektroniczne) sieci stacji benzynowych i supermarketów. W latach 30-tych do 50-tych używane były blaszane karty Charge-Plate (Rysunek 1). Oprócz blaszanych, używane były również karty papierowe i kartonowe, podatne na uszkodzenia.

Rysunek 1. Przykładowa karta Charge-Plate [30]

Rozwój kart plastikowych rozpoczął się w 1950 roku w Stanach Zjednoczonych. Niska cena tworzywa sztucznego PCW pozwoliła na produkcję kart plastikowych, które były bardziej trwałe od kart używanych wcześniej, a jednocześnie nadawały się do codziennego użytku. Pierwszą plastikową kartę wydał Diners Club w 1950 roku. Następnymi organizacjami, które zajmowały się kartami

plastikowymi były Visa i MasterCard. Dzięki temu karty stały się popularne najpierw w USA, a następnie opanowały również całą Europę i pozostałą część świata. Dzisiaj można z nich skorzystać praktycznie na całym świecie. W różnych krajach można płacić jedną kartą kredytową, dzięki czemu nie ma konieczności zakupu obcej waluty przed wyjazdem za granicę. Rocznie wydawane są setki milionów kart [6].

Początkowo funkcje kart były bardzo proste. Głównym zadaniem było przechowywanie danych i zabezpieczanie ich przed fałszowaniem. Wśród zapisanych danych było imię i nazwisko posiadacza oraz numer karty. Wiele kart posiadało dodatkowo pole na podpis posiadacza karty. W pierwszych kartach zabezpieczenie przed fałszerstwem zależało głównie od osoby przyjmującej taką kartę – mogłaby nie sprawdzić podpisu posiadacza i przyjąć skradzioną kartę. Później, jako zabezpieczenie, zaczęto stosować osobisty numer identyfikacyjny (PIN).

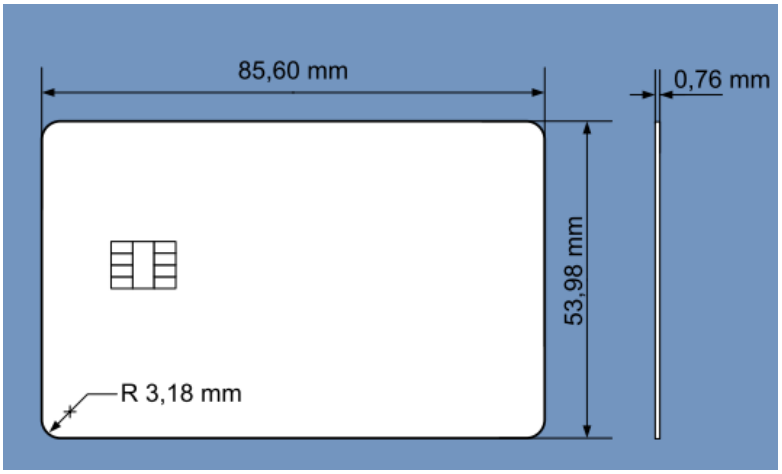
Do dzisiaj używanych jest kilka rodzajów kart plastikowych. Jednym z nich są karty z paskiem magnetycznym, który łatwo odczytać kopiując w ten sposób dane karty. W 1970 roku postęp w elektronice pozwolił na umieszczenie układu krzemowego wewnątrz karty. W 1974 roku ceny układów krzemowych były akceptowalne [6]. Zaczęto wtedy masowo produkować karty elektroniczne. W 1994 roku została opublikowana pierwsza wersja specyfikacji kart EMV (standard kart płatniczych Europay, MasterCard i VISA). Aktualnie karty są bardziej rozbudowane. Jedna karta może spełniać wiele funkcji. Przykładem jest Elektroniczna Legitymacja Studencka (ELS), która jest m.in. identyfikatorem studenta, kartą biblioteczną i biletem miesięcznym komunikacji miejskiej.

2.2. Budowa

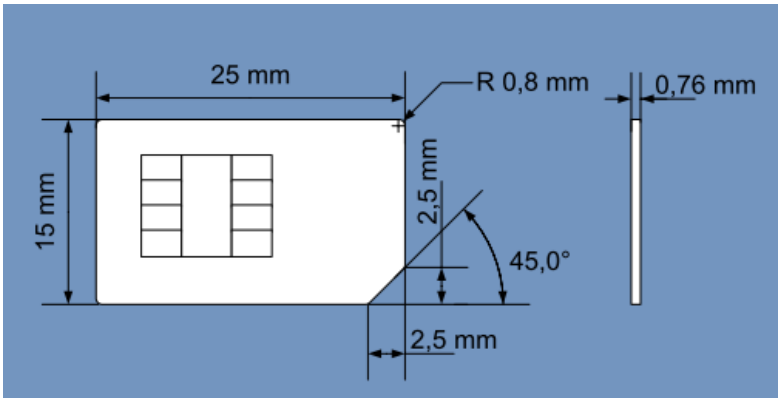
W rozdziale tym zostały przedstawione formaty kart zgodne z normą ISO/IEC 7810 [14] oraz opis układu elektronicznego.

2.2.1. Formaty kart

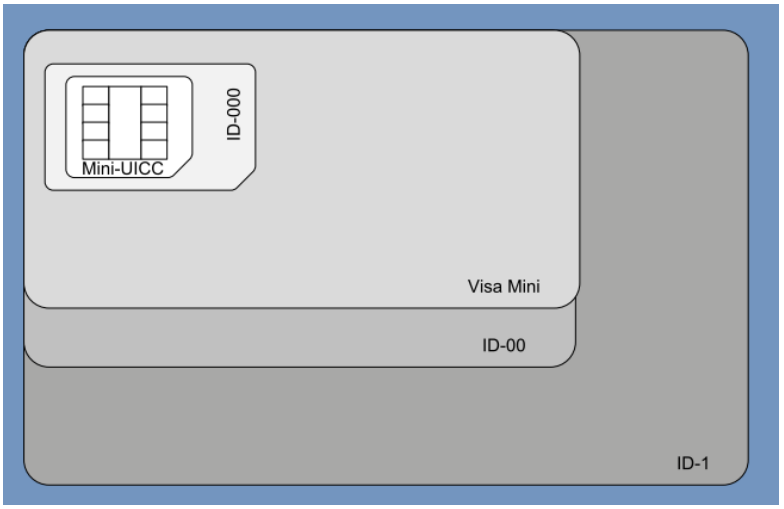
Norma ISO/IEC 7810 [14] uwzględnia kilka formatów, najczęściej spotykane to ID-1 (najpopularniejsze karty płatnicze, Rysunek 2) oraz ID-000 (karty SIM, Rysunek 3). Rysunek 4 przedstawia porównanie formatów kart elektronicznych.



Rysunek 2. Wymiary karty formatu ID-1 [4]



Rysunek 3. Wymiary karty formatu ID-000 [4]

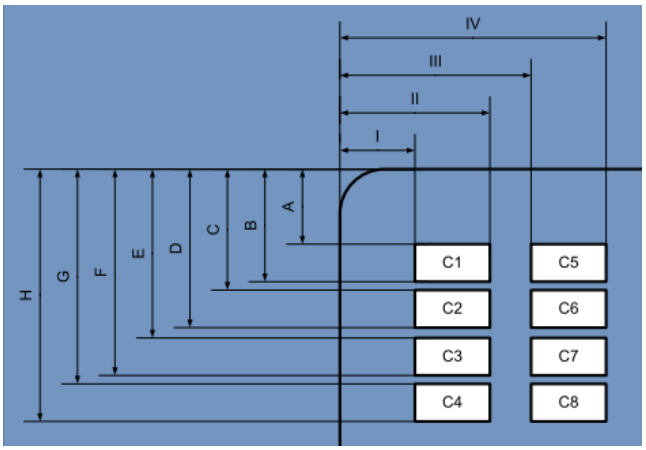


Rysunek 4. Porównanie formatów kart elektronicznych [4]

2.2.2. Układ elektroniczny

Układ elektroniczny to najważniejszy element karty elektronicznej. To w nim znajduje się procesor oraz pamięć. Układ elektroniczny może korzystać z interfejsu stykowego oraz bezstykowego.

W przypadku interfejsu stykowego na karcie elektronicznej widoczne są styki. Ich wymiary oraz pozycja na karcie są bardzo dokładnie określone w normie ISO/IEC 7816-2 [15] (Rysunek 5).



Rysunek 5. Wymiarowanie styków na karcie elektronicznej [4]

Wymiar	Odległość [mm]
I	<= 10,25
II	>= 12,25
III	<= 17,87
IV	>= 19,87
A	<= 19,23
B	>= 20,93
C	<= 21,77
D	>= 23,47
E	<= 24,31
F	>= 26,01
G	<= 26,85
H	>= 28,55

Pomimo tak dokładnej specyfikacji producenci różnych układów przygotowują styki o różnych kształtach (Rysunek 6). Nawet styki w kartach jednego producenta mogą się różnić między sobą. Niemniej spełniają one wymagania normy.

Rysunek 6. Różne kształty styków [31]

Układy z interfejsem bezstykowym wyposażone są w antenę zatopioną wewnątrz karty, natomiast nie posiadają styków. Istnieją również karty elektroniczne posiadające oba interfejsy. Zostały one szczegółowo opisane w rozdziale 2.4.

2.3. Komunikacja

Norma ISO/IEC 7816-3 [16] opisuje dwa, najczęściej stosowane protokoły komunikacji z kartami elektronicznymi (procesorowymi). Są one oznaczone jako T=0 oraz T=1. W normie ISO/IEC 14443-4 [13] został opisany protokół T=CL używany podczas komunikacji z kartami z interfejsem bezstykowym (np. karty dualne).

Transmisja jest dwukierunkowa, lecz jedynie w trybie półdupleksu. Rodzaj protokołu ustalany jest obustronnie po włożeniu karty do czytnika (w pracy występuje zamiennie określenie terminal, oba oznaczają urządzenie podłączone do komputera, pośredniczące w komunikacji pomiędzy kartą elektroniczną i komputerem) i przesłaniu bajtów odpowiedzi na resetowanie (ATR, ang. *Answer To Reset*) w przypadku komunikacji poprzez interfejs stykowy oraz bajtów ATQA (ang. *Answer To Request*) w przypadku użycia interfejsu bezstykowego.

2.3.1. Protokoły komunikacyjne

Protokół T=0

W protokole T=0 jednostką transmisji jest jeden bajt. Błędy transmisji są wykrywane na poziomie jednego bajtu poprzez dodatkowy bit parzystości. Do karty są przesyłane polecenia w jednostce danych protokołu aplikacji (APDU ang. *Application Protocol Data Unit*). Ich struktura jest przedstawiona w tabeli 1.

Tabela 1. Struktura polecenia APDU w protokole T=0

CLA	**INS**	**P1**	**P2**	P3	Dane

Pola oznaczone niebieskim kolorem są obowiązkowe – muszą występować w każdym poleceniu. Jeśli polecenie nie zawiera bajta P3 przyjmowana jest dla niego wartość 0. Pole to oznacza liczbę bajtów przesyłanych w polu danych, zatem wartość 0 oznacza brak pola danych. Po każdym poleceniu wysłanym do karty karta odsyła odpowiedź. Jej struktura jest przedstawiona poniżej. Pole CLA oznacza klasę polecenia – różne klasy poleceń dla komunikacji szyfrowanej i nieszyfrowanej. Pole INS to kod numeryczny instrukcji. Pola P1 i P2 są parametrami dla instrukcji. Pole dane zawiera dodatkowe informacje zależne od polecenia. Zostały one dokładniej opisane w części poświęconej poleceniom APDU (rozdział 2.3.2).

Tabela 2. Struktura odpowiedzi APDU w protokole T=0

Dane	**SW1**	**SW2**

Tabela 2 przedstawia strukturę odpowiedzi. Pola oznaczone niebieskim kolorem występują w każdej odpowiedzi. Jeśli pole z danymi nie jest przesyłane, lecz dane są gotowe do przesłania, wtedy w polu

SW2 jest przesyłana liczba bajtów danych, które można odczytać. Odczytu dokonuje się wysyłając polecenie GET RESPONSE (pole P3 zawiera liczbę bajtów, które mają zostać odczytane, jest ono równe wartości z pola SW2 ostatniej odpowiedzi). Odpowiedź na to polecenie składa się z wszystkich trzech pól.

Przykładem kart korzystających z protokołu T=0 są karty SIM.

Protokół T=1

Protokół T=1 jest zorientowany blokowo co oznacza, że najmniejszą jednostką przesyłanych danych jest jeden blok. Tabela 3 przedstawia strukturę bloku.

Tabela 3. Struktura bloku TPDU w protokole T=1

Prolog			Zawartość	Epilog
Adres węzła (NAD)	Bajt sterujący protokołem (PCB)	Długość (LEN)	APDU	Suma kontrolna (EDC)
1 bajt	1 bajt	1 bajt	0-254 bajtów	1 lub 2 bajty

Błędy transmisji wykrywane są po przesłaniu całego bloku, na podstawie sumy kontrolnej przesłanej w polu EDC. Pole LEN zwiera długość pola z zawartością (APDU). Może być ono wyzerowane, zatem pole APDU jest opcjonalne.

W odróżnieniu od protokołu T=0 odpowiedź na polecenie APDU automatycznie zawiera pole danych. Nie ma konieczności wysyłania polecenia GET RESPONSE. Protokół T=1 może posiadać dodatkowe pole w poleceniu APDU, oznaczone jako Le. Jest ono przesyłane po polu danych i oznacza długość pola danych, które powinno zostać przesłane w odpowiedzi. Jest to wykorzystywane np. podczas odczytu plików na karcie, gdzie oznacza ile bajtów z danego pliku ma zostać odczytanych.

Przykładem kart korzystających z protokołu T=1 są legitymacje studenckie personalizowane na kartach Oberthur Cosmo.

Protokół T=CL

Protokół T=CL jest używany podczas komunikacji z kartami z wykorzystaniem interfejsu bezstykowego. Jest to protokół blokowy (struktura bloku analogiczna do struktury bloku w protokole T=1). Norma ISO/IEC 14443-4 [13] opisuje szczegóły dotyczące protokołu T=CL.

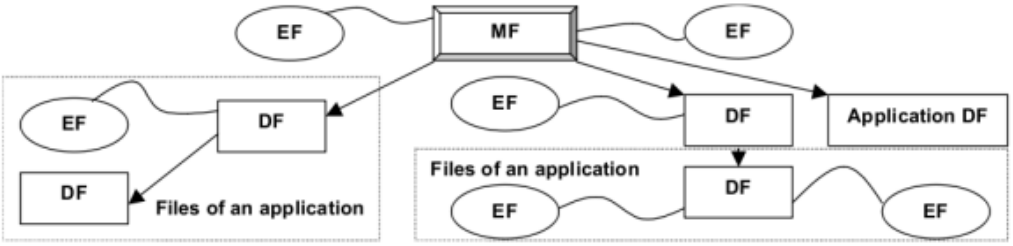
2.3.2. Struktura plików

Norma ISO/IEC 7816-4 [17] opisuje system plików w kartach elektronicznych. Jest to struktura drzewiasta. Głównym plikiem jest plik MF (ang. *Master File*), w którym znajdują się pliki przechowujące dane (EF, ang. *Elementary File*) oraz pliki DF (ang. *Dedicated File*), które przechowują aplikacje, pliki EF oraz inne pliki DF.

Pliki EF dzielą się na dwie kategorie. Pierwszą z nich stanowią pliki transparentne. Nie posiadają one określonej struktury, dostęp do plików tego rodzaju odbywa się na poziomie bajtów.

Przykładem plików transparentnych są pliki EF.CERT oraz EF.ELS znajdujące się w Elektronicznej Legitymacji Studenckiej. Druga grupa to pliki rekordowe. Posiadają określoną strukturę rekordu, dostęp jest możliwy jedynie do wybranego rekordu. Przykładem pliku rekordowego jest plik książki telefonicznej zapisany w pamięci karty SIM.

Rysunek 7 przedstawia przykładową strukturę plików w karcie elektronicznej.



Rysunek 7. Przykładowa struktura plików w karcie elektronicznej [17]

2.3.3. Skrócony opis poleceń APDU

Poniżej przedstawiono skrócony opis poleceń wg normy ISO/IEC 7816-4 [17].

READ BINARY

Polecenie służy do odczytywania zawartości wyselekcjonowanego pliku transparentnego. Wartość pola INS = 0xB0. Pola P1 i P2 oznaczają nr bajta w pliku, od którego należy zacząć odczyt. Pole Le oznacza liczbę bajtów do odczytania.

WRITE BINARY

Polecenie służy do zapisu podanych bajtów do wyselekcjonowanego pliku. Pole INS = 0xD0. Pola P1 i P2 oznaczają nr bajta w pliku, od którego należy zacząć zapis. Pole Lc oznacza liczbę bajtów do zapisania w pliku. Pole dane zawiera kolejne bajty do zapisania w pliku.

UPDATE BINARY

Polecenie służy do zaktualizowania bajtów wyselekcjonowanego pliku. Pole INS = 0xD6. Pola P1 i P2 oznaczają nr bajta w pliku, od którego należy zacząć zapis. Pole Lc oznacza ilość bajtów do zapisania w pliku. Pole dane zawiera kolejne bajty do zapisania w pliku.

ERASE BINARY

Polecenie służy do logicznego wyczyszczenia bajtów w wyselekcjonowanym pliku. Pole INS = 0x0E. Pola P1 i P2 oznaczają nr bajtu w pliku, od którego należy zacząć czyszczenie. Pole Lc może przyjąć wartość 0x00 lub 0x02 i oznacza ilość bajtów przesyłanych w polu danych. Pole dane zawiera numer pierwszego bajta, który nie zostanie wyczyszczony (wartość ta musi być większa od wartości w polach P1, P2).

READ RECORD

Polecenie służy do odczytu wyselekcjonowanego pliku rekordowego. Pole INS = 0xB2. Pole P1 oznacza numer rekordu, który ma zostać odczytany (0x00 – rekord bieżący), pole P2 oznacza sposób odczytu bajtów. Pole Le oznacza liczbę bajtów do odczytania.

WRITE RECORD

Polecenie służy do zapisu wartości z pola danych do wyselekcjonowanego pliku rekordowego. Pole INS = 0xD2. Pole P1 oznacza numer rekordu, który ma zostać zapisany (0x00 – rekord bieżący), pole P2 oznacza sposób zapisu bajtów. Pole Lc oznacza długość pola danych. Pole danych zawiera wartości do zapisania w pliku.

APPEND RECORD

Polecenie dodaje rekord na końcu pliku rekordowego o strukturze liniowej lub zapisuje rekord na pozycji 1. w pliku o strukturze cyklicznej. Pole INS = 0xE2, P1 = 0x00, P2 = 0x00. Pole Lc zawiera długość pola danych. Pole danych zawiera rekord do zapisania w pliku.

UPDATE RECORD

Polecenie analogiczne do polecenia WRITE RECORD. Pole INS = 0xDC.

GET DATA

Polecenie służy do odczytu podstawowych obiektów danych. Pole INS = 0xCA. Pola P1 i P2 dokładnie zostały opisane w normie [17]. Pole Le oznacza liczbę bajtów spodziewanych w odpowiedzi.

PUT DATA

Polecenie służy do zapisu podstawowych obiektów danych. Pole INS = 0xDA. Pola P1 i P2 dokładnie zostały opisane w normie [17]. Pole Le oznacza liczbę bajtów spodziewanych w odpowiedzi.

SELECT FILE

Polecenie uaktywnia wybrany plik EF lub DF. Pole INS = 0xA4. Pola P1 i P2 określają sposób wyboru pliku. Pole Lc zawiera długość pola danych. W polu danych znajduje się identyfikator pliku.

VERIFY

Polecenie uwierzytelnia użytkownika karty (np. hasło, nr PIN). Pola INS = 0x20, P1 = 0x00. Pole P2 wskazuje identyfikator klucza, który zostanie użyty do uwierzytelnienia. Pole Lc zawiera długość pola danych. Pole danych zawiera wartość klucza.

INTERNAL AUTHENTICATE

Polecenie umożliwiające uwierzytelnienie użytkownika na podstawie wartości otrzymanej poleceniem GET CHALLENGE oraz klucza. Pole INS = 0x88. Pole P1 zależy od algorytmu

wspieranego przez kartę. Pole P2 oznacza numer klucza. Pole Lc oznacza długość pola danych. Pole danych zawiera wartość obliczoną na podstawie wartości odczytanej przez GET CHALLENGE zgodnie z wybranym algorytmem. Pole Le określa maksymalną długość pola danych w odpowiedzi.

EXTERNAL AUTHENTICATE

Polecenie umożliwiające obustronne uwierzytelnienie na podstawie wartości otrzymanej poleceniem GET CHALLENGE oraz klucza. Pole INS = 0x82. Pole P1 zależy od algorytmu wspieranego przez kartę. Pole P2 oznacza numer klucza. Pole Lc oznacza długość pola danych. Pole danych zawiera wartość obliczoną na podstawie wartości odczytanej przez GET CHALLENGE zgodnie z wybranym algorytmem.

GET CHALLENGE

Polecenie wysyłane przed uwierzytelnieniem w celu uzyskania wartości wejściowej do algorytmu uwierzytelniającego. Pole INS=0x82. Pola P1, P2 = 0x0000. Pole Le określa maksymalną długość pola danych w odpowiedzi.

GET RESPONSE

Polecenie używane w protokole T=0 w celu odczytania odpowiedzi po wykonaniu polecenia. Pole INS = 0xC0. Pola P1, P2 = 0x0000. Pole Le określa maksymalną długość pola danych w odpowiedzi.

ENVELOPE

Polecenie do przesyłania innych poleceń APDU lub ich części, które nie mogłyby zostać normalnie przesłane przy użyciu wybranego protokołu. Pole INS = 0xC2. Pola P1, P2 = 0x0000. Pole Lc określa długość pola danych. Pole danych zawiera polecenie APDU. Pole Le określa maksymalną długość pola danych w odpowiedzi.

2.4. Klasyfikacja kart

Karty można podzielić ze względu na różne kryteria. W dalszej części rozdziału omówiono podział uwzględniający następujące kryteria: rodzaj układu elektronicznego, nośnik danych, zastosowanie karty oraz sposób nadruku danych osobowych. Od tego momentu, ilekroć w pracy występuje stwierdzenie karta dotyczy ono karty elektronicznej (ang. *Smart Card*), o ile nie zaznaczono inaczej.

2.4.1. Podział ze względu na zastosowany układ elektroniczny

Ze względu na zastosowany układ wyróżniamy dwa rodzaje kart: pamięciowe oraz mikroprocesorowe.

- Pamięciowe

  Karty tego rodzaju umożliwiają bezpośredni dostęp do pamięci poprzez wyprowadzone styki. Zasilanie jest dostarczane podczas operacji odczytu i zapisu, natomiast sama karta nie posiada wbudowanego źródła zasilania. Aby dane zapisane w pamięci karty nie zostały utracone wykorzystuje się pamięć nieulotną (EEPROM). Przykładem są przedpłacone karty telefoniczne oraz karty pacjenta stosowane w Niemczech od 1990 roku [6].

- Mikroprocesorowe

  Karty mikroprocesorowe w odróżnieniu od pamięciowych posiadają procesor. Umożliwia on wykonywanie operacji kryptograficznych (np. zabezpieczenie komunikacji) ale zmienia on również dostęp do pamięci. Karty te posiadają system operacyjny, który zarządza systemem plików zatem nie ma bezpośredniego dostępu do pamięci z zewnątrz. Karty procesorowe pozwalają na samodzielne tworzenie i wgrywanie oprogramowania. Przykładem są karty SIM, karty z podpisem elektronicznym, elektroniczna portmonetka.

2.4.2. Podział ze względu na nośnik danych

Ze względu na nośnik danych można wyróżnić trzy rodzaje kart: magnetyczne, elektroniczne oraz optyczne.

- Magnetyczne

  Nośnikiem danych jest pasek magnetyczny. Karty tego rodzaju można w prosty sposób kopiować a same dane zawarte na pasku magnetycznym zmienić, przy czym trudno później udowodnić modyfikację.

- Elektroniczne

  Dane są przechowywane w pamięci wewnętrznej karty. Karty te dzielą się m. in. na pamięciowe i procesorowe. Szczegółowy podział kart elektronicznych został przedstawiony w dalszej części rozdziału.

- Optyczne

  Karty z pamięcią optyczną korzystają z technologii używanej na płytach CD. Mogą one pomieścić kilka megabajtów danych.

2.4.3. Podział ze względu na zastosowanie

Najpopularniejsze zastosowania kart zostały przedstawione poniżej. Większość dotyczy kart elektronicznych, jednakże np. karty lojalnościowe nie muszą być kartami elektronicznymi.

- Lojalnościowe
  Karty służące do zbierania punktów, karty rabatowe w sklepach, na stacjach paliw itp.

- Dostępowe
  Karty służące jako klucze, umożliwiające dostęp do pomieszczeń, np. karty pokojowe w hotelach, karty pracownicze w Politechnice Poznańskiej.

- Sieci teleinformatycznych
  Karty umożliwiające dostęp do sieci teleinformatycznych (karty SIM).

- Kryptograficzne
  Karty zawierające certyfikaty, umożliwiające elektroniczne podpisywanie dokumentów.

- Komunikacyjne
  Karty służące jako bilety miesięczne w środkach transportu publicznego (KomKarta, PEKA).

- Płatnicze
  Karty płatnicze wydawane przez banki, elektroniczna portmonetka (PrePaid).

- Identyfikacyjne
  Karty identyfikujące osobę (legitymacje studenckie, karty biblioteczne, karty pracownicze, planowany nowy dowód osobisty).

- Inne
  Oprócz wymienionych wyżej istnieją jeszcze inne zastosowania kart elektronicznych (np. karty telefoniczne).

Możliwość tworzenia aplikacji działających w środowisku karty elektronicznej powoduje, że możliwości kart elektronicznych są bardzo duże. Coraz częściej poszczególne funkcjonalności kart elektronicznych są ze sobą łączone (np. legitymacja studencka wraz z kartą biblioteczną, PEKA łącząca funkcje karty komunikacyjnej oraz karty płatniczej w postaci elektronicznej portmonetki).

2.4.4. Podział ze względu na sposób naniesienia danych posiadacza

Karty płatnicze zawierają podstawowe dane karty oraz posiadacza naniesione na powierzchnię karty. Ze względu na sposób nanoszenia tych danych wyróżnia się karty płaskie oraz tłoczone.

- Płaskie
  Karty płaskie posiadają nadrukowane (lub wypalone laserem) dane posiadacza, numer karty oraz datę ważności.

- Tłoczone

  Karty tłoczone, jak sama nazwa wskazuje, posiadają wytłoczone dane posiadacza, nr karty oraz datę ważności.

2.4.5. Podział kart elektronicznych

Karty elektroniczne różnią się między sobą m. in. używanymi interfejsami komunikacyjnymi oraz rodzajem systemu operacyjnego. Poniżej przedstawiono podział kart elektronicznych ze względu na te parametry.

- Ze względu na interfejs
  - Stykowe

    Komunikacja z kartą odbywa się poprzez zestaw styków umieszczonych na karcie. Zasilanie do karty dostarczane jest również poprzez styki. Pozycjonowanie styków na karcie jest ściśle określone.

  - Bezstykowe

    Komunikacja z kartą odbywa się bezkontaktowo, zasięg działania jest ograniczony do kilku centymetrów. Wewnątrz karty zatopiona jest antena. Karty bezstykowe występują również w formie np. naklejek na telefon.

  - Hybrydowe

    Karty hybrydowe posiadają przynajmniej dwa nośniki informacji (mogą to być np. elektroniczny układ z interfejsem stykowym i elektroniczny układ z interfejsem bezstykowym, co ważne, oba układy są niezależne od siebie).

  - Dualne

    Karty dualne posiadają dwa interfejsy dostępowe – stykowy oraz bezstykowy, które są połączone z jednym układem elektronicznym.

- Ze względu na system operacyjny
  - Natywne

    Karty natywne są projektowane w określonym celu. Posiadają zamknięty system operacyjny, najczęściej zainstalowaną jedną aplikację a możliwości tworzenia oprogramowania na kartę są ograniczone.

  - Wieloaplikacyjne

    Karty z systemami operacyjnymi JavaCard, Multos, .net, BasicCard. Tworzenie oprogramowania na karty tego rodzaju jest proste – można używać popularnych języków Java, C#, Basic.

2.5. Podsumowanie

Karty elektroniczne coraz częściej pojawiają się w codziennym życiu. Ich funkcjonalności są bardzo zróżnicowane. Występuje duża różnorodność nie tylko interfejsów i protokołów, ale nawet karty korzystające z tego samego interfejsu i protokołu mogą się różnić od siebie na poziomie poleceń APDU. W takich przypadkach różnice pojawiają się w polach parametrów poleceń. Niestety nie istnieje jedna aplikacja dla wszystkich kart elektronicznych.

3. ZAJĘCIA LABORATORYJNE

Rozdział ten został opracowany na podstawie [3].

Celem przedmiotu o nazwie „Programowanie kart elektronicznych" jest przedstawienie zastosowań kart elektronicznych w różnych dziedzinach życia codziennego oraz zdobycie praktycznych umiejętności projektowania systemów wykorzystujących karty elektroniczne.
Program przedmiotu obejmuje następujące zagadnienia:

- historia i budowa kart elektronicznych,
- standardy, specyfikacje i normy,
- platforma GlobalPlatform,
- systemy operacyjne: Java, BasicCard, Multos, .NET
- systemy płatności EMV,
- karty SIM,
- personalizacja legitymacji studenckich,
- karty z interfejsem bezkontaktowym - RFID,
- karty elektroniczne i PKI (bezpieczeństwo, certyfikaty),
- zastosowanie kart elektronicznych: paszport, opieka zdrowotna, transport.

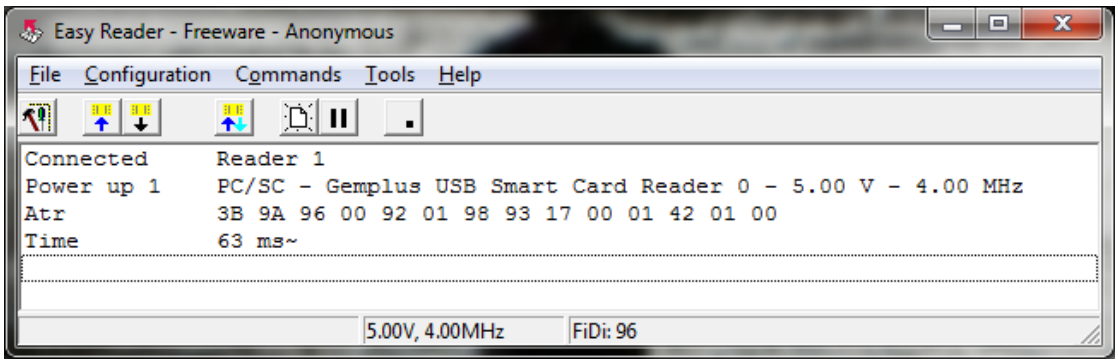
Opis używanego sprzętu

Laboratorium wyposażone jest w komputery z wbudowanymi czytnikami kart elektronicznych z interfejsem stykowym oraz zewnętrzne czytniki z interfejsem stykowym i bezstykowym podłączane do portu USB. Umożliwiają one przeprowadzenie wszystkich zaplanowanych zajęć laboratoryjnych. Potrzebne wyposażenie laboratorium stanowią także karty elektroniczne, które zostają wypożyczone studentom na czas odpowiednich zajęć. Są to karty SIM różnych operatorów, czyste karty JavaCard, BasicCard, .Net, testowe legitymacje studenckie oraz karty EMV.

Opis używanego oprogramowania

Obsługa różnych kart elektronicznych wymaga odpowiedniego oprogramowania, które jest zainstalowane na komputerach w laboratorium. Poniżej przedstawiona jest skrócona lista oprogramowania używanego podczas zajęć oraz omówione funkcje oprogramowania, z których korzystają studenci w trakcie zajęć.

- Edsi Easy Reader [18]



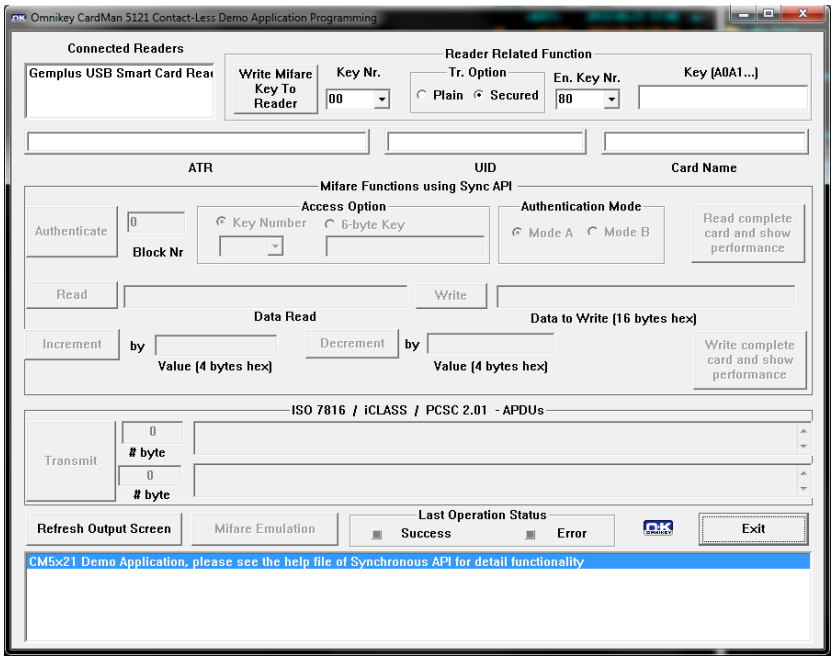
Rysunek 8. Edsi Easy Reader - główne okno programu

Oprogramowanie udostępniane przez producenta nieodpłatnie, umożliwia przesyłanie poleceń APDU z czytnika do karty. Posiada funkcję odczytu nr ATR z wyjaśnieniem poszczególnych bajtów. Historię przesłanych poleceń można zapisać do pliku.

Funkcje oferowane przez ten program zostały zaimplementowane we wtyczkach APDU oraz ATR do programu SmartCard Suite 2.0.

- Omnikey Diagnostic Tool [19]

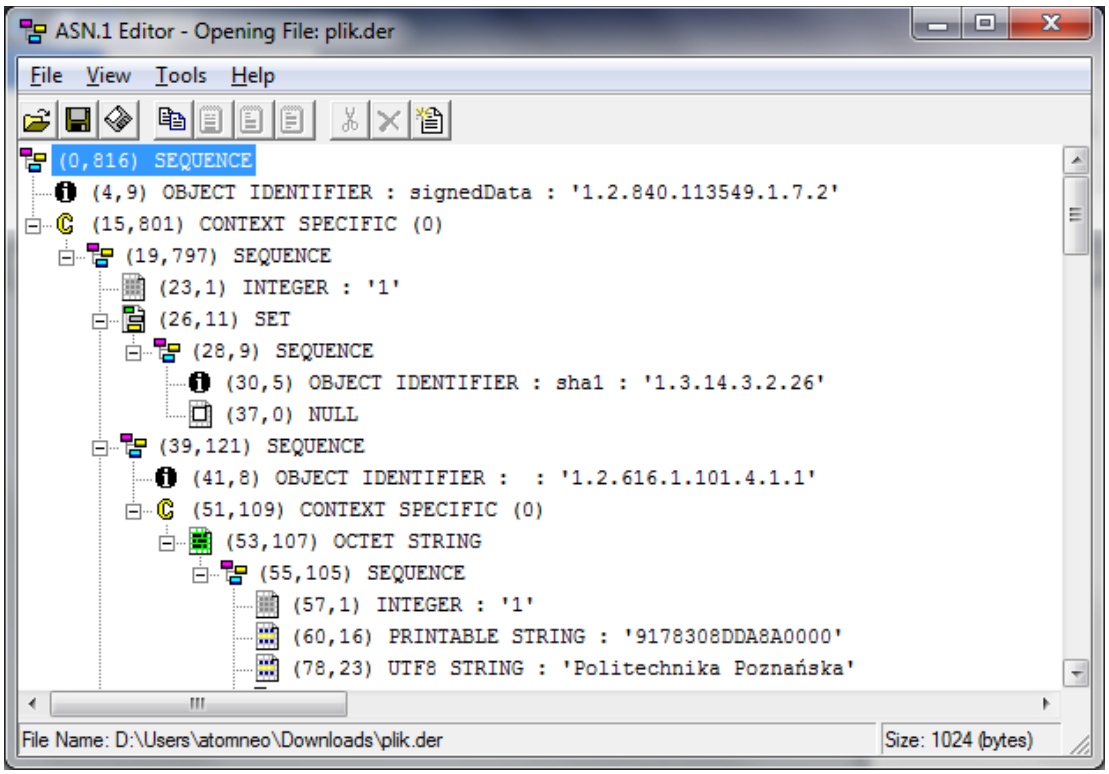
Oprogramowanie służące do odczytu i zapisu bloków i sektorów w pamięci MIFARE. Realizuje również funkcje zapewniające uwierzytelnianie konieczne do uzyskania dostępu do bloków pamięci MIFARE. Funkcje te obsługuje wtyczka MIFARE w programie SmartCard Suite 2.0.



Rysunek 9. Wygląd ekranu programu Omnikey Diagnostic Tool

- ASN.1 Editor [20]

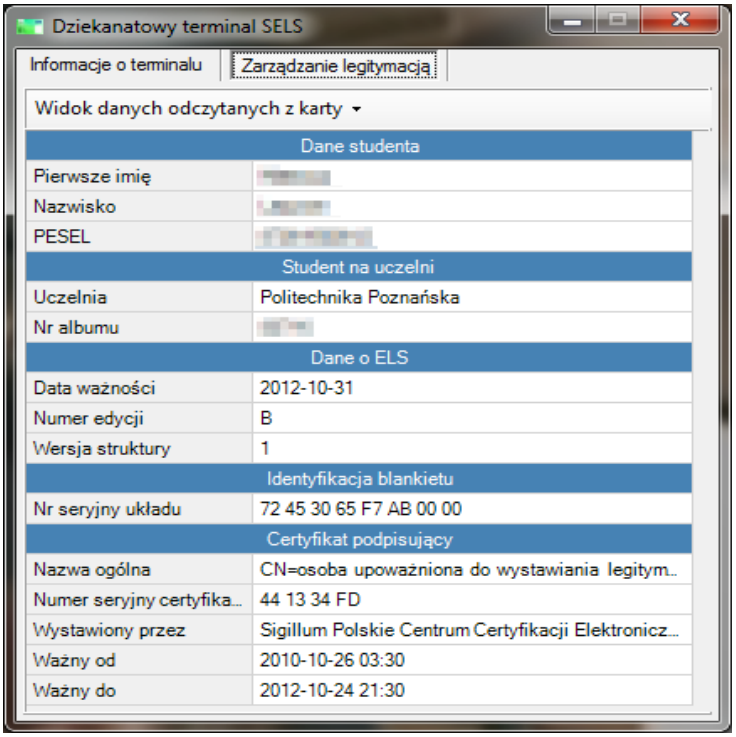
  Program wyświetlający strukturę danych zapisanych w notacji ASN.1 (dane zapisane w ELS). Funkcjonalność ta została zaimplementowana we wtyczce ASN.1 i jest wykorzystywana we wtyczce ELS.



Rysunek 10. Główne okno programu z otwartym przykładowym plikiem

- SELSTerminal Offline

  Oprogramowanie stworzone na potrzeby Międzyuczelnianego Centrum Personalizacji Legitymacji Studenckich przy Politechnice Poznańskiej do obsługi legitymacji w dziekanatach (niedostępna publicznie). Aplikacja umożliwia odczyt danych z Elektronicznej Legitymacji Studenta. Wyświetla dane zapisane w pamięci karty. Jest to zaimplementowane we wtyczce ELS, która dodatkowo umożliwia odczyt danych z elektronicznych legitymacji doktoranckich (legitymacja jest rozpoznawana automatycznie).



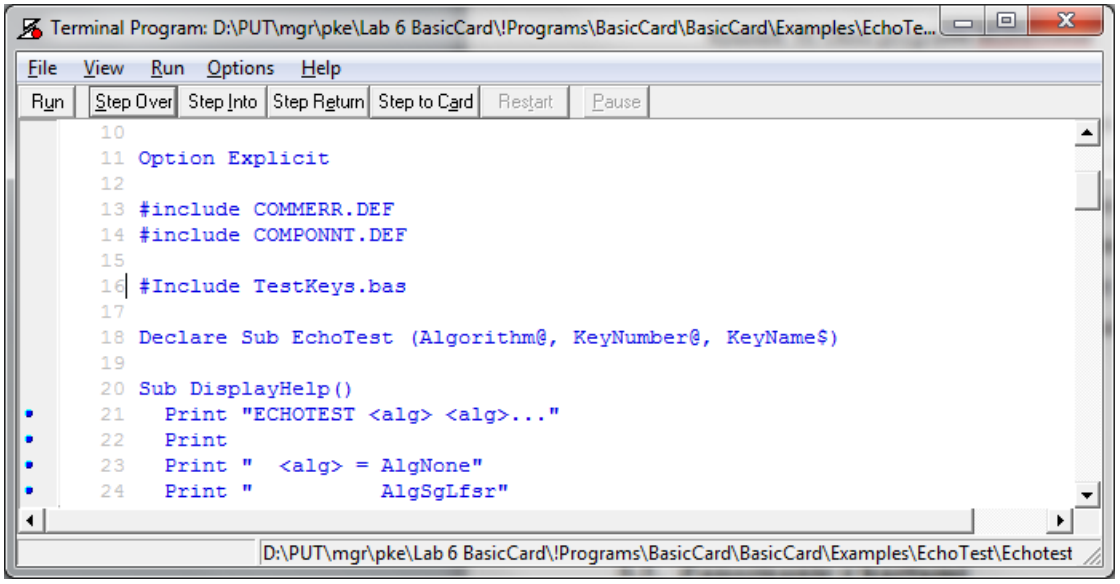
Rysunek 11. Okno programu SELSTerminal

- Oberthur Application Loader [21]

  Aplikacja obsługująca karty z systemem operacyjnym JavaCard, służy do instalowania aplikacji Java w pamięci karty. Instalacja aplikacji w środowisku kartowym jest zależna m. in. od producenta karty oraz od wersji oprogramowania znajdującego się w pamięci ROM karty. Jest to utrudnieniem podczas tworzenia uniwersalnego instalatora aplikacji. Wtyczka JavaCard Installer pozwala na instalację stworzonego oprogramowania Java na kartach wyprodukowanych przez firmę Oberthur Technologies. Oprogramowanie niedostępne publicznie.

- ZeitControl BasicCard [22]

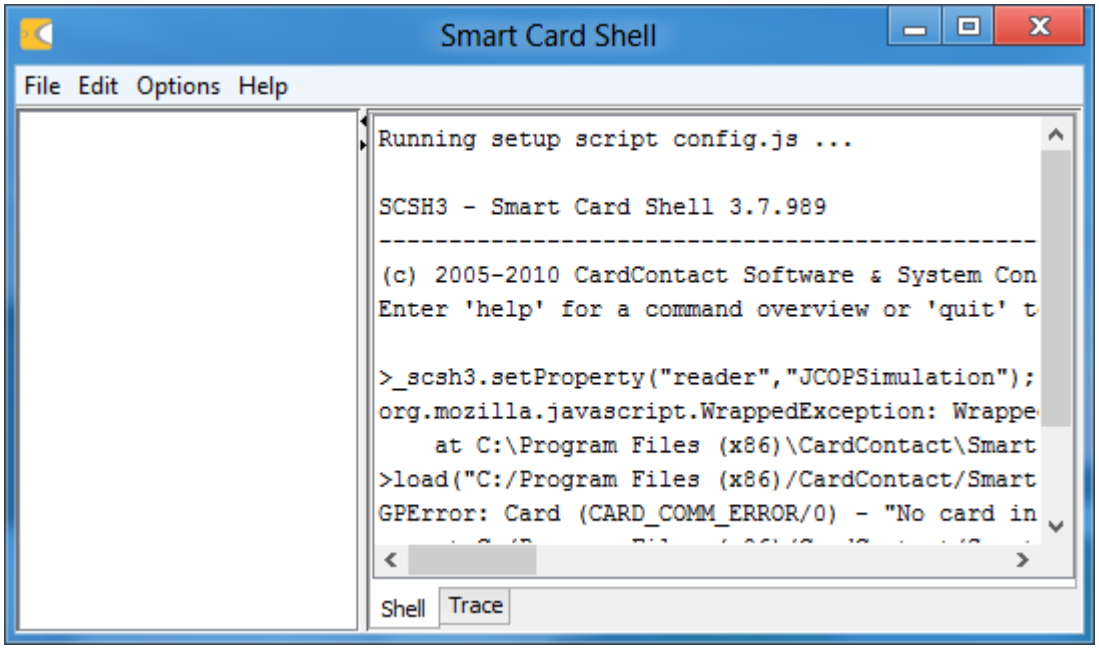
  Aplikacja umożliwiająca tworzenie, testowanie oraz instalowanie w pamięci karty oprogramowania BasicCard. Jest to środowisko programistyczne w związku z czym wtyczka do programu SmartCard Suite 2.0 o zbliżonej funkcjonalności nie została przygotowana. Podobnie do przypadku JavaCard instalacja oprogramowania na różnych kartach BasicCard może się różnić.



Rysunek 12. Ekran aplikacji ZeitControl Basic Card z wczytanym przykładowym programem

- Smart Card Shell [23]

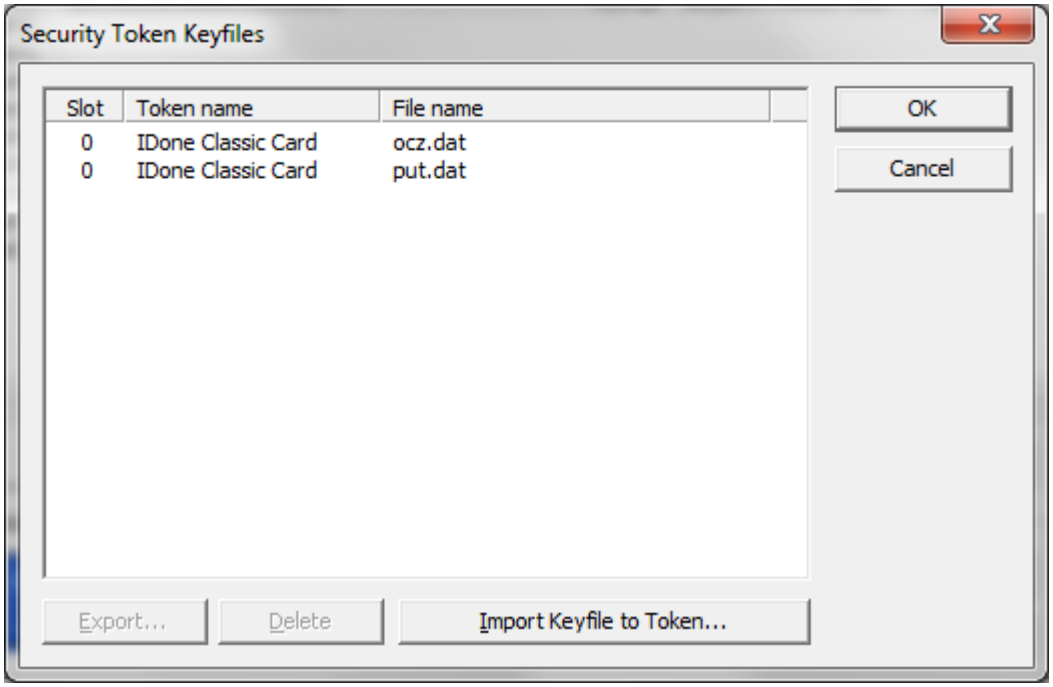
  Środowisko umożliwiające tworzenie skryptów korzystających z kart elektronicznych. W oprogramowaniu SmartCard Suite 2.0 nie zostało to zaimplementowane, jednakże istnieje możliwość przygotowania odpowiedniej wtyczki obsługującej język skryptowy.



Rysunek 13. Główne okno programu Smart Card Shell

- True Crypt [24]

  Oprogramowanie służące do szyfrowania danych, tworzy wirtualne dyski twarde w plikach oraz umożliwia szyfrowanie fizycznych napędów. Wykorzystanie kart elektronicznych ogranicza się do przechowywania kluczy (tokenów) w pamięci karty.



Rysunek 14. Okno programu True Crypt z listą tokenów zapisanych na karcie

- Oberthur AuthentIC [21]

  Oprogramowanie udostępniające biblioteki wymagane przez inne programy (True Crypt, Thunderbird) w celu korzystania z certyfikatów i kluczy zapisanych na karcie. Dodatkowo umożliwia zmianę kodu PIN karty. Oprogramowanie nie jest dostępne publicznie.

Rysunek 15. Okno programu AuthentIC wyświetlające informacje o karcie

3.1. Przedstawienie scenariuszy zajęć oraz warunków zaliczenia przedmiotu

Rozdział ten opisuje pierwsze zajęcia z przedmiotu laboratorium kart elektronicznych, na których zostaną przedstawione warunki zaliczenia przedmiotu.

3.1.1. Warunki zaliczenia laboratorium programowania kart elektronicznych

Na ocenę końcową z przedmiotu laboratorium programowania kart elektronicznych składają się:

- Aktywność na zajęciach
  Aktywność oceniana jest na podstawie wykonanych w ramach zajęć punktowanych zadań. Każde poprawnie wykonane zadanie jest punktowane maksymalnie 1 punktem. Na podstawie zdobytej liczby punktów wystawiana jest ocena dla tej części zajęć.

- Test obejmujący materiał przedstawiony na zajęciach
  Test obejmuje zagadnienia przedstawione w ramach zajęć. Składa się z losowo wybranych pytań dotyczących każdego ćwiczenia. Każda poprawna odpowiedź punktowana jest 1 punktem.

- Projekt praktyczny lub teoretyczny
  Projekt stanowi trzeci składnik oceny końcowej. Po przeprowadzeniu wszystkich zaplanowanych zajęć laboratoryjnych studenci wykonują projekt, który następnie jest oceniany.

Ocena końcowa jest wypadkową wszystkich trzech części, przy czym warunkiem koniecznym jest uzyskanie pozytywnej oceny z każdej z nich.

3.2. Zapoznanie z kartami

3.2.1. Cel ćwiczenia

Ćwiczenie ma na celu zapoznanie studentów z podstawowymi zagadnieniami dotyczącymi kart elektronicznych. W trakcie ćwiczenia zostaną przedstawione podstawowe właściwości i funkcje kart elektronicznych oraz konfiguracja popularnych aplikacji do obsługi kart elektronicznych.

3.2.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Dostęp do Internetu.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Oberthur AuthentIC WebPack.

- Oprogramowanie TrueCrypt.

- Wymagane przygotowanie studenta
  - Legitymacja studencka.
  - Dostęp do systemu eLogin PP.
  - Umiejętność skonfigurowania klienta poczty.

3.2.3. Przebieg ćwiczenia

Na początku ćwiczenia należy uaktywnić na legitymacjach studenckich (każdy student na własnej) aplikację PKCS#11. Uaktywnienie następuje zgodnie z instrukcją opublikowaną na stronie *elogin.put.poznan.pl*. Pomyślna aktywacja jest konieczna do wykonania dalszych zadań w ćwiczeniu. Następnie przeprowadzone zostaną dyskusje na tematy podpisu elektronicznego, szyfrowania danych oraz podpisywania dokumentów. Po każdej z dyskusji został przewidziany czas na wykonanie zadań.

3.2.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Aktywacja aplikacji PKCS#11 na legitymacji studenckiej.
2. Skonfigurowanie klienta poczty do obsługi kart elektronicznych.
3. Wysłanie podpisanej i zaszyfrowanej wiadomości do prowadzącego.
4. Skonfigurowanie oprogramowania TrueCrypt.
5. Odszyfrowanie pliku kluczem znajdującym się w pamięci karty.
6. Podpisanie i przesłanie podpisanych plików do prowadzącego.

3.2.5. Przykłady zadań, pytań sprawdzających wiedzę

- Czym jest oprogramowanie „middleware”?
- Co musi się znajdować w pamięci karty, aby można było ją użyć do podpisywania wiadomości?
- Co jest potrzebne do zaszyfrowania wiadomości?
- Co jest potrzebne do podpisania wiadomości?
- Kiedy można podpisać dokument pdf?
- Kiedy oprogramowanie TrueCrypt jest gotowe do korzystania z plików kluczy zapisanych na karcie?
- W jaki sposób w PKI można uzyskać certyfikat?

3.3. Elektroniczna Legitymacja Studenta

3.3.1. Cel ćwiczenia

Ćwiczenie ma na celu przedstawienie studentom systemu plików kart elektronicznych na przykładzie Elektronicznej Legitymacji Studenckiej. W trakcie ćwiczenia studenci zostaną zapoznani

z numerem ATR karty, ze strukturą plików w ELS oraz ze sposobem niskopoziomowego dostępu do kart elektronicznych – wybieranie i odczyt plików przy użyciu przesyłania poleceń i odbierania odpowiedzi APDU.

3.3.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie ASN.1 Editor lub SmartCard Suite 2.0 z wtyczką ASN.1.
  - Oprogramowanie Edsi EasyReader lub SmartCard Suite 2.0 z wtyczką APDU.
  - Dokumentacja: [16], [17], [19].
- Wymagane przygotowanie studenta
  - Elektroniczna legitymacja studencka.

3.3.3. Przebieg ćwiczenia

Ćwiczenie rozpoczyna się od wyjaśnienia, czym jest ATR karty, po czym studenci analizują jego kolejne wartości. Następnie zostanie przedstawiony sposób odczytu zawartości plików na karcie oraz sposób zaprezentowania go w formie graficznej (drzewo), po czym studenci samodzielnie wykonają zadania.

3.3.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Zinterpretowanie wartości ATR.
2. Odczytanie plików EF.ELS i EF.CERT z ELS.
3. Zinterpretowanie danych zawartych w plikach EF.ELS i EF.CERT.

3.3.5. Przykłady zadań, pytań sprawdzających wiedzę

- Czym jest wartość ATR?
- Z czego składa się polecenie READ BINARY?
- Jak wygląda struktura ELS?
- Jak działa polecenie READ BINARY?
- Jak działa polecenie SELECT FILE?
- W jakiej normie jest zdefiniowana składnia polecenia READ BINARY?
- W jakiej normie jest zdefiniowana składnia polecenia SELECT FILE?
- W jakiej normie jest opisana interpretacja wartości ATR karty?
- Jakie bajty występują w notacji ASN.1?
- Co oznacza TLV w notacji ASN.1?

- Jak wyglądają bajty statusu po poprawnym wykonaniu polecenia SELECT FILE dla protokołu T=1?
- Co otrzymamy w odpowiedzi na polecenie READ BINARY 00 B0 00 00 80?
- Co otrzymamy w odpowiedzi na polecenie READ BINARY 00 B0 01 02 03?

3.4. Interfejs bezstykowy – pamięć MIFARE

3.4.1. Cel ćwiczenia

Ćwiczenie ma na celu zapoznanie studentów z możliwościami najpopularniejszych kart elektronicznych z interfejsem bezstykowym w jaki zostały wyposażone ELS. Studenci poznają organizację i zasady bezpieczeństwa w pamięci MIFARE 1K.

3.4.2. Wymagania

- Wymagania dot. stanowiska
  - Bezstykowy czytnik kart elektronicznych.
  - Testowe legitymacje studenckie z zapisaną określoną wartością w pamięci MIFARE.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Omnikey Diagnostic Tool lub SmartCard Suite 2.0 z wtyczką MIFARE.
  - Oprogramowanie Edsi EasyReader lub SmartCard Suite 2.0 z wtyczką APDU.
  - Dokumentacja: [13], [18].
  - Wymagane przygotowanie studenta
  - Elektroniczna legitymacja studencka.
  - Umiejętność konwersji pomiędzy systemem dziesiętnym i szesnastkowym.

3.4.3. Przebieg ćwiczenia

Ćwiczenie rozpoczyna się wprowadzeniem do organizacji pamięci MIFARE, zostanie przedstawiona jej struktura, informacje o zabezpieczeniach oraz o sposobie dostępu do pamięci MIFARE. Następnie studenci wykonują zadania. Na czas zajęć studenci otrzymają uprawnienia do otwierania drzwi sali laboratoryjnej legitymacją studencką w celach demonstracyjnych.

3.4.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Odczytanie nr UID legitymacji (każdy student własnej) i przesłanie prowadzącemu – na tej podstawie zostaną przydzielone uprawnienia do otwierania drzwi.
2. Odczytanie sektorów pamięci MIFARE w poszukiwaniu określonego ciągu wartości oraz przesłanie nr UID karty i nr sektora, w którym poszukiwany ciąg się znajduje.

3. Protokół T=CL – wykonanie zadań z ćwiczenia dot. ELS przy wykorzystaniu interfejsu bezkontaktowego.

3.4.5. Przykłady zadań, pytań sprawdzających wiedzę

- Jaka wartość jest zapisana w sektorze 0. bloku 0. pamięci MIFARE?
- Z czego składa się sektor pamięci MIFARE 1K?
- Z ilu bajtów składa się klucz umożliwiający odczyt zawartości sektora pamięci MIFARE?
- Dla jakich kart jest możliwe odczytanie zawartości pamięci MIFARE poprzez interfejs stykowy?

3.5. Karty SIM

3.5.1. Cel ćwiczenia

Ćwiczenie ma na celu przedstawienie organizacji karty SIM, sposobu komunikacji z kartą oraz wskazanie, jak odczytywać i zapisywać dane z karty SIM. Zostanie przedstawione znaczenie niektórych plików znajdujących się w pamięci karty SIM.

3.5.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Karty SIM (najlepiej różnych operatorów) w rozmiarze ID-1.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Edsi EasyReader lub SmartCard Suite 2.0 z wtyczką APDU.
  - Oprogramowanie SmartCard Suite 2.0 z wtyczkami SIM Phonebook oraz SIM Manager.
  - Dokumentacja: [12].
- Wymagane przygotowanie studenta
  - Wykonanie wcześniejszych ćwiczeń – rozumienie zasady wysyłania poleceń i odczytu odpowiedzi APDU.

3.5.3. Przebieg ćwiczenia

Prowadzący przedstawi studentom podstawowe informacje o protokole T=0, z którego korzystają karty SIM. Następnie w formie dyskusji zostaną przedstawione kolejne pliki. Po każdej z dyskusji został przewidziany czas na wykonanie zadań.

3.5.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Uwierzytelnienie kodem PIN.
2. Odczytanie numeru IMSI.

3. Odczytanie nr telefonu przypisanego do karty SIM.
4. Odczytanie numerów telefonów zapisanych w poszczególnych komórkach pamięci karty SIM.
5. Zapisanie swojego kontaktu w 16 (0x10) komórce pamięci.
6. Przesłanie odczytanych danych prowadzącemu.

3.5.5. Przykłady zadań, pytań sprawdzających wiedzę

- Jak wygląda struktura karty SIM?
- W jakim pliku są przechowywane informacje o numerach telefonów?
- W jakim pliku jest przechowywany numer seryjny karty?
- Jakiego polecenia należy użyć, aby uzyskać informacje o pliku podczas komunikacji karty zgodnej z protokołem T=0?
- W jaki sposób można uzyskać informacje zwrotne podczas komunikacji karty zgodnej z protokołem T=1?
- W jaki sposób są zapisywane numery telefonów na karcie?
- W jaki sposób jest zapisywany opis w książce telefonicznej?

3.6. Karty płatnicze EMV

3.6.1. Cel ćwiczenia

Ćwiczenie ma na celu przedstawienie organizacji karty EMV, sposobu komunikacji z kartą oraz wskazanie, jak odczytywać i zapisywać dane z karty EMV. Zostanie przedstawione znaczenie niektórych plików znajdujących się w pamięci karty EMV.

3.6.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Edsi EasyReader lub SmartCard Suite 2.0 z wtyczką APDU.
  - Dokumentacja: [8], [9], [10], [11].
- Wymagane przygotowanie studenta
  - Wykonanie wcześniejszych ćwiczeń – rozumienie zasady wysyłania poleceń i odczytu odpowiedzi APDU.

3.6.3. Przebieg ćwiczenia

Prowadzący przedstawi studentom podstawowe informacje o kartach EMV, wyjaśni okoliczności powstania specyfikacji. Następnie w formie dyskusji zostaną przedstawione kolejne pliki oraz polecenia APDU. Po każdej z dyskusji został przewidziany czas na wykonanie zadań.

3.6.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Odczyt plików, do których jest dostęp i analiza odczytanych wartości.
2. Odczytanie nazwy aplikacji, preferowanych języków, numeru karty oraz daty ważności.

3.6.5. Przykłady zadań, pytań sprawdzających wiedzę

- Co oznacza skrót EMV?
- Jakich kart dotyczy specyfikacja EMV?
- Jak przebiega płatność?
- Jakie są modele autoryzacji posiadacza karty, terminala?

3.7. Międzyuczelniane Centrum Personalizacji Legitymacji Studenckich

3.7.1. Cel ćwiczenia

Ćwiczenie ma na celu przedstawienie pracy MCP oraz procesu personalizacji legitymacji studenckiej – poznanie cyklu produkcji ELS.

3.7.2. Wymagania

- Wymagania dot. stanowiska
  - Umówienie wizyty w MCP.
- Wymagane przygotowanie studenta
  - Obecność w MCP.

3.7.3. Przebieg ćwiczenia

Ćwiczenie odbywa się w MCP, studenci ze względów organizacyjnych podzieleni na grupy nie większe niż 15 osobowe wejdą do MCP i poznają proces personalizacji ELS. Podczas ćwiczenia zostaną zadane pytania, na które odpowiedzi należy przesłać prowadzącemu.

3.7.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Przesłanie odpowiedzi na pytania zadane przez prowadzącego.

3.7.5. Przykłady zadań, pytań sprawdzających wiedzę

- Jaka taśma jest wykorzystywana do personalizacji ELS?
- Na czym polega druk termosublimacyjno-retransferowy?
- Na czym polega druk termosublimacyjny?
- Jaki jest kolejny etap po personalizacji części stykowej karty?

- Jaki jest ostatni etap personalizacji ELS wykonywany w drukarkach?
- Ile czasu trwa personalizacja jednej ELS?
- Jakie czynności odbywają się w drukarkach MCP?

3.8. Karty BasicCard

3.8.1. Cel ćwiczenia

Ćwiczenie ma na celu wprowadzenie studentów do tematyki kart elektronicznych zarządzanych systemem BasicCard. Studenci zostaną zapoznani z podstawami tworzenia aplikacji w języku Basic, sposobami wgrywania aplikacji na kartę i korzystania z tych aplikacji.

3.8.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie ZeitControl BasicCard.
  - Oprogramowanie umożliwiające przesyłanie poleceń APDU.
  - Przykładowe programy BasicCard.
  - Karty BasicCard.
  - Dokumentacja: [25].
- Wymagane przygotowanie studenta
  - Wykonanie wcześniejszych ćwiczeń – rozumienie zasad wysyłania poleceń i odbioru odpowiedzi APDU.
  - Zapoznanie się z podstawami składni języka BasicCard.

3.8.3. Przebieg ćwiczenia

Studenci przychodzą na zajęcia z podstawową wiedzą o składni języka BasicCard. Na zajęciach zostaną przedstawione najważniejsze elementy języka BasicCard, następnie studenci stworzą aplikację HelloWorld, która najpierw zostanie przetestowana w terminalu a następnie, jeśli nie będzie błędów, aplikacja zostanie wgrana na kartę i zweryfikowanie przy pomocy wcześniej poznanych metod przesyłania poleceń APDU i otrzymywania odpowiedzi. Następnie studenci stworzą aplikację dla tokenu, czyli urządzenia z wyświetlaczem, na którym po włożeniu karty elektronicznej portmonetki wyświetlane jest saldo.

3.8.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Przygotowanie aplikacji HelloWorld.

2. Przygotowanie aplikacji dla tokenu.
3. Przesłanie kodu aplikacji prowadzącemu.

3.8.5. Przykłady zadań, pytań sprawdzających wiedzę

- Jaki jest ostatni stan karty BasicCard, z którego nie można przejść już do innego stanu?
- Pomiędzy jakimi stanami karty BasicCard przejścia są dozwolone?
- Jak jest definiowana zmienna przechowywana w pamięci stałej karty?
- Jak jest deklarowana zmienna dostępna wewnątrz i w czasie wykonywania procedury?
- Jakiego typu zmienne są dostępne globalnie?
- Jakiego typu zmienne są dostępne lokalnie?
- Jak jest definiowany blok BasicCard określający sposób komunikacji karty z otoczeniem?
- Co musi zawierać oprogramowanie dla BalanceReader'a?
- Jakie są wartości CLA i INS polecenia BalanceReader'a?

3.9. Global Platform i JavaCard

3.9.1. Cel ćwiczenia

Ćwiczenie ma na celu przedstawienie środowiska Global Platform oraz wprowadzenie studentów do programowania aplikacji w języku Java dla kart JavaCard.

3.9.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Oberthur Application Loader lub SmartCard Suite 2.0 z wtyczką Installer.
  - Przykładowe programy JavaCard.
  - Karty JavaCard.
  - Dokumentacja: [20].
- Wymagane przygotowanie studenta
  - Wykonanie wcześniejszych ćwiczeń – rozumienie zasad wysyłania poleceń i odbioru odpowiedzi APDU.
  - Zapoznanie się z podstawami składni języka JavaCard.

3.9.3. Przebieg ćwiczenia

Prowadzący przedstawi środowisko Global Platform oraz sposób instalacji aplikacji w środowisku GP. Następnie zostanie przedstawiony kod przykładowej aplikacji, która zostanie zainstalowana w pamięci

karty. Po przeanalizowaniu działania przykładowej aplikacji studenci utworzą samodzielnie aplikację HelloWorld.

3.9.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Instalacja przygotowanego apletu w pamięci karty.
2. Stworzenie apletu HelloWorld i instalacja w pamięci karty.
3. Przesłanie prowadzącemu kodu źródłowego stworzonego apletu.

3.9.5. Przykłady zadań, pytań sprawdzających wiedzę

- Na co pozwala GlobalPlatform?
- Jak powinien zostać utworzony aplet JavaCard?
- Jakiego typu zmiennej nie powinno się używać w konstrukcji pętli?
- Jakie pliki ładuje się do pamięci karty podczas wgrywania apletu JavaCard?

3.10. Karty .Net

3.10.1. Cel ćwiczenia

Ćwiczenie ma na celu wprowadzenie studentów do tematyki kart elektronicznych zarządzanych systemem .Net. Studenci zostaną zapoznani z podstawami tworzenia aplikacji w języku C#, sposobami wgrywania aplikacji na kartę i korzystania z tych aplikacji.

3.10.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Visual Studio wraz z zainstalowaną wtyczką .Net Card Explorer (znajduje się w sterownikach Gemalto).
  - Dokumentacja: [32].
- Wymagane przygotowanie studenta
  - Wykonanie wcześniejszych ćwiczeń – rozumienie zasad wysyłania poleceń i odbioru odpowiedzi APDU.
  - Zapoznanie się z podstawami składni języka C#.

3.10.3. Przebieg ćwiczenia

Na zajęciach zostaną przedstawione najważniejsze elementy języka C# związane z obsługą kart elektronicznych, następnie studenci stworzą aplikację HelloWorld. Aplikacja zostanie wgrana na kartę

i zweryfikowana przy pomocy wcześniej poznanych metod przesyłania poleceń APDU lub aplikacji klienckiej.

3.10.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Przygotowanie aplikacji HelloWorld.
2. Przesłanie kodu aplikacji prowadzącemu.

3.10.5. Przykłady zadań, pytań sprawdzających wiedzę

- W jakiej architekturze tworzy się oprogramowanie dla kart .Net?
- Jaki komponent jest odpowiedzialny za pobieranie aplikacji wysyłanej do karty?

3.11. Skrypty w Smart Card Shell

3.11.1. Cel ćwiczenia

Ćwiczenie ma na celu zapoznanie studentów z tworzeniem skryptów wykorzystujących karty elektroniczne.

3.11.2. Wymagania

- Wymagania dot. stanowiska
  - Stykowy czytnik kart elektronicznych.
  - Oprogramowanie Adobe Reader.
  - Oprogramowanie Smart Card Shell.
  - Przykładowe skrypty w Smart Card Shell.
- Wymagane przygotowanie studenta
  - Wykonanie wcześniejszych ćwiczeń.
  - Zapoznanie się z podstawami składni skryptów w Smart Card Shell.
  - Wymagania są zależne od ćwiczenia, które będzie opracowywane.

3.11.3. Przebieg ćwiczenia

Prowadzący przedstawia studentom oprogramowanie Smart Card Shell, w którym studenci stworzą skrypty wykonujące działania z wcześniejszych ćwiczeń.

3.11.4. Zadania do wykonania

W ramach ćwiczenia studenci powinni wykonać następujące zadania:

1. Przygotowanie skryptu na podstawie jednego z wcześniejszych ćwiczeń.
2. Przesłanie kodu skryptu z opisem jego funkcjonalności i komentarzami.

3.12. Projekt

Projekt ma na celu utrwalenie oraz poszerzenie wiedzy poznanej na zajęciach. Studenci przygotowują projekt teoretyczny w postaci prezentacji lub praktyczny w postaci oprogramowania. Do dyspozycji pozostają komputery w laboratorium, lecz nie jest wymagane korzystanie z nich. Po wykonaniu projektu odbywa się jego prezentacja. Realizacja niektórych projektów wymaga dostępu do poufnych danych, w związku z czym wymagane jest podpisanie klauzuli o zachowaniu poufności.

Przykładowe tematy projektów:

- Oprogramowanie do przesyłania poleceń i odbierania odpowiedzi APDU.
- Oprogramowanie do odczytu i zapisu pamięci MIFARE.
- Oprogramowanie testujące klucze sektorów pamięci MIFARE (brute force).
- Karta PEKA – prezentacja.
- Karty EMV – prezentacja.
- Pl.id – nowe dowody osobiste, prezentacja.

4. PROJEKT APLIKACJI

Oprogramowanie SmartCard Suite 2.0 jest kolejną wersją aplikacji stworzonej w ramach projektu z przedmiotu Laboratorium programowania kart elektronicznych. Pierwsza wersja posiadała wiele ograniczeń i często występowały błędy w działaniu po wyjęciu karty z czytnika. Wersja druga została zaprojektowana od podstaw a jej funkcjonalność znacząco rozbudowana.

Jak wspomniano we wcześniejszych rozdziałach, w użyciu jest wiele kart elektronicznych różnego rodzaju. Karty komunikują się z wykorzystaniem różnych protokołów. Składnia tych samych poleceń może się nieco od siebie różnić. Uniemożliwia to stworzenie uniwersalnej aplikacji obsługującej wszystkie karty. Problem ten został rozwiązany poprzez wprowadzenie systemu wtyczek, które w łatwy sposób można rozbudowywać oraz tworzyć nowe, wprowadzając obsługę wybranej karty do programu Smart Card Suite 2.0.

4.1. Wymagania funkcjonalne

Wymagania funkcjonalne zostały podzielone na wymagania funkcjonalne głównej aplikacji oraz wymagania funkcjonalne pozostałych wtyczek.

Główna aplikacja

- Udostępnienie interfejsu do komunikacji z kartą dla modułów.
- Logowanie przesyłanych poleceń do karty (i odpowiedzi).
- Umożliwienie wyboru czytnika i odczyt ATR aktywnej karty.
- Przechwytywanie zdarzeń włożenia i wyjęcia karty.

Wtyczka APDU

- Wysyłanie polecenia podanego w formie bajtowej.
- Wysyłanie polecenia z listy poleceń.
- Wyświetlanie i podstawowa interpretacja odpowiedzi karty.

Wtyczka ATR

- Odczyt i wyświetlenie numeru ATR karty.
- Wyświetlenie znaczenia poszczególnych bajtów.

Wtyczka MIFARE

- Wyświetlenie numeru UID karty.
- Załadowanie kluczy.
- Uwierzytelnienie do sektora.

- Odczyt sektorów.
- Zapis sektorów.

Wtyczka ELS

- Odczytanie zawartości plików EF.ELS i EF.CERT z karty.
- Przekazanie danych do wtyczki wyświetlającej strukturę ASN.1.

Wtyczka ASN.1

- Wyświetlenie struktury danych zapisanych w formacie ASN.1.

Wtyczka SIM

- Zarządzanie kodem PIN.
- Edycja książki telefonicznej.
- Odczyt zawartości plików IMSI i MSISDN.

Wtyczka Terminal Listing

- Wyświetlenie podłączonych terminali.
- Oznaczenie aktywnego terminala.

4.2. Wymagania pozafunkcjonalne

Wymagania pozafunkcjonalne zostały przedstawione w kilku grupach: funkcjonalność, niezawodność, użyteczność, wydajność, łatwość konserwacji oraz przenośność.

Funkcjonalność

- Główna aplikacja powinna poprawnie współpracować z wtyczkami.
- Wtyczki powinny mieć możliwość komunikacji pomiędzy sobą.
- Powinna być możliwa rozbudowa oprogramowania.

Niezawodność

- W przypadku wystąpienia przewidzianego błędu program wyświetla informację o nim.
- W przypadku wystąpienia nieprzewidzianego błędu program wyświetla ogólną informację.
- Jeżeli błąd nie jest krytyczny aplikacja powinna dalej poprawnie funkcjonować.

Użyteczność

- Użytkowanie oprogramowania powinno być intuicyjne dla użytkowników.

- Powinna istnieć możliwość dostosowania interfejsu użytkownika.
- Podstawowe funkcje wtyczek powinny być dostępne po jednym kliknięciu.

Wydajność

- W interfejsie użytkownika operacje trwające poniżej 3 sekund nie muszą być poprzedzane komunikatem.
- Operacje dłuższe powinny być poprzedzone komunikatem i wskazywać bieżący postęp operacji.
- Oprogramowanie nie powinno w nadmiernym stopniu zużywać zasobów komputera.

Łatwość konserwacji

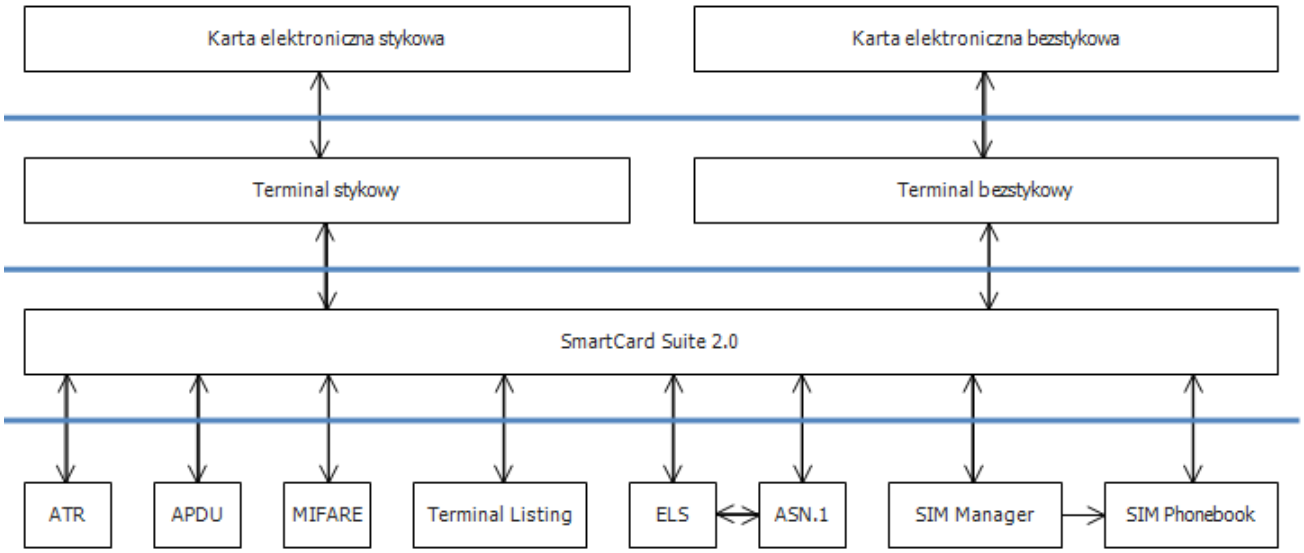
- Powinien być dostępny kod źródłowy z komentarzami.
- Oprogramowanie powinno mieć architekturę modułową.

Przenośność

- Oprogramowanie powinno działać na różnych systemach operacyjnych rodziny Windows (koniecznie na Windows 7 32 bit oraz 64 bit a także Windows 8 i Windows XP).

4.3. Architektura

Aplikacja SmartCard Suite 2.0 jest zbudowana w architekturze modułowej. Moduły, które są widoczne dla użytkownika to wspomniany we wstępie rozdziału system wtyczek. Oprogramowanie korzysta z dodatkowego, niewidocznego dla użytkownika modułu, umożliwiającego bezproblemową komunikację pomiędzy główną aplikacją i wtyczkami. Rysunek 16 przedstawia architekturę oprogramowania wraz z przygotowanym zestawem wtyczek.



Rysunek 16. Architektura oprogramowania SmartCard Suite 2.0

Na powyższym rysunku wyróżnić można cztery poziomy architektury. Poziom pierwszy stanowią różnego rodzaju karty elektroniczne korzystające z interfejsu stykowego oraz bezstykowego. Drugi poziom to czytniki tych kart, pośredniczące w komunikacji pomiędzy komputerem i kartą. Na trzecim poziomie znajduje się główna aplikacja, która bezpośrednio komunikuje się jedynie z terminalami oraz z zainstalowanymi wtyczkami. Główna aplikacja zarządza całą komunikacją pomiędzy wtyczkami i czytnikiem kart elektronicznych. Ostatni poziom stanowią same wtyczki. To w nich znajduje się logika odpowiedzialna za obsługę poszczególnych rodzajów kart elektronicznych.

4.4. Główna aplikacja

Najważniejszym założeniem projektu była możliwość łatwej rozbudowy głównej aplikacji poprzez system wtyczek. Z tego powodu główna aplikacja powinna wyszukiwać pliki zgodnych wtyczek, które użytkownik może uruchomić. Podczas uruchamiania aplikacji jest skanowany podkatalog *Plugins* w poszukiwaniu plików z rozszerzeniem *dll*. Po sporządzeniu listy plików aplikacja określa, które z nich zawierają klasy implementujące interfejs *IPlugin*. Na tej podstawie jest tworzona ostateczna lista wtyczek, która zostanie wyświetlona użytkownikowi.

Gdy odpowiednia wtyczka zostanie uruchomiona, jej okno pojawi się na ekranie umożliwiając korzystanie z możliwości wybranej wtyczki. Domyślnie okno wtyczki jest zadokowane w centralnej części głównej aplikacji, lecz użytkownik może dowolnie okno to przenosić.

Główna aplikacja daje wtyczkom dostęp do zainstalowanych w systemie czytników kart oraz wspiera komunikację pomiędzy wtyczkami udostępniając listę załadowanych wtyczek i możliwość przesłania polecenia do wtyczki znajdującej się na tej liście.

4.5. Wtyczki

Poniższa lista przedstawia wtyczki do głównej aplikacji wraz z krótkim opisem ich funkcjonalności.

- APDU

  Wtyczka umożliwia wysyłanie poleceń APDU do karty i odbieranie odpowiedzi. Umożliwia zapisanie historii przesyłanych poleceń i odpowiedzi.
- ATR

  Wtyczka wyświetla numer ATR karty oraz analizuje jego znaczenie.
- MIFARE

  Wtyczka wyświetla numer UID karty oraz umożliwia odczyt i zapis poszczególnych bloków i sektorów pamięci MIFARE z wykorzystaniem uwierzytelniania.
- ELS

  Wtyczka odczytuje zawartość plików EF.CERT i EF.ELS, przekazuje je do wtyczki ASN.1, po czym wyświetla dane zawarte w tych plikach. Obsługuje również Elektroniczne legitymacje doktoranckie. Rozpoznanie rodzaju legitymacji odbywa się automatycznie.
- ASN.1

  Wtyczka wyświetla strukturę plików zapisanych w formacie ASN.1 oraz udostępnia funkcje dla wtyczki ELS.
- SIM

  Wtyczka wyświetla numer telefonu przypisany do karty (o ile znajduje się na karcie), umożliwia zarządzanie książką telefoniczną i kodem PIN.
- Terminal Listing

  Wtyczka wyświetla listę czytników oraz zaznacza aktywny.

5. IMPLEMENTACJA

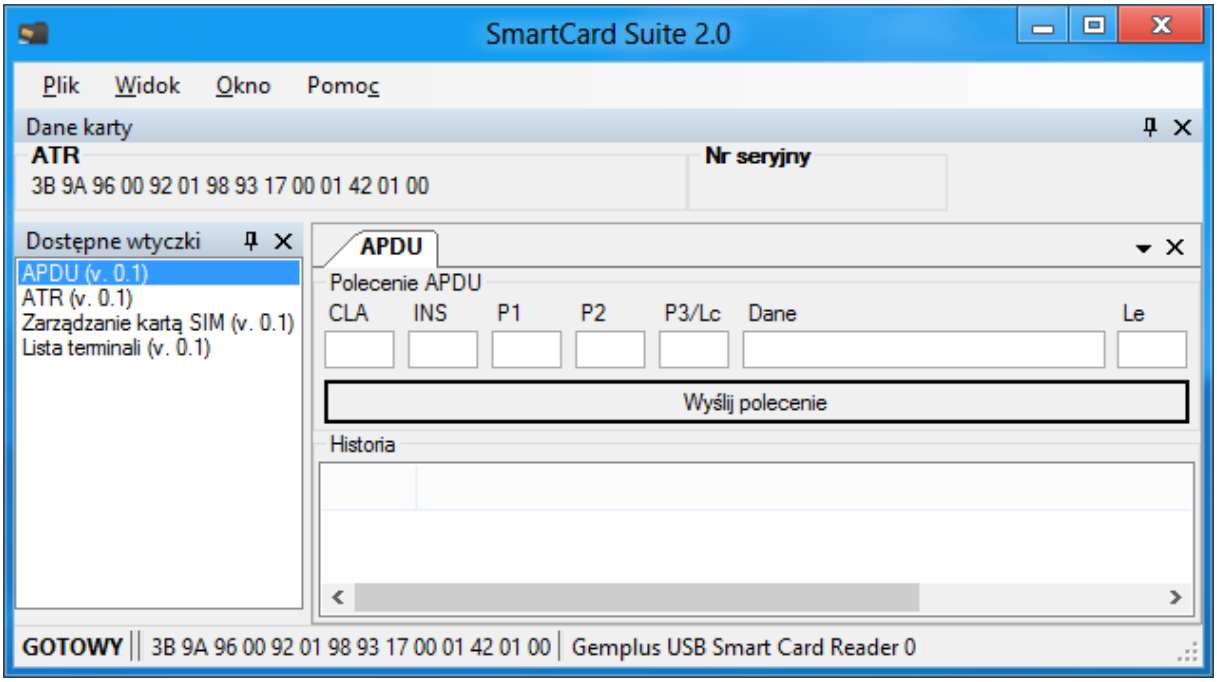
Poniższy rozdział zawiera szczegółowy opis implementacji ważniejszych fragmentów aplikacji. Aplikacja została przetestowana w systemach Windows 7 32-bit oraz 64-bit. Niektóre funkcje zostały przetestowane dodatkowo w systemie Windows 8 64-bitowym.

5.1. Implementacja głównej aplikacji

Aplikacja do uruchomienia wymaga środowiska .NET Framework w wersji 3.5. Jest ona dostępna zarówno dla systemu Windows 7 jak i dla wersji XP oraz 8.

5.1.1. Graficzny interfejs użytkownika

Główna aplikacja została stworzona w oparciu o interfejs MDI. Okno programu składa się z paska menu zawierającego polecenia umożliwiające konfigurację interfejsu oraz paska stanu, na którym są wyświetlane podstawowe informacje o aktualnie używanej karcie, terminalu i interfejsie. Centralną część okna zajmuje pole, w którym wyświetlane są okna wtyczek. Rysunek 17 przedstawia okno aplikacji z uruchomioną wtyczką APDU. W górnej części, pod menu wyświetlone są dane karty, natomiast z lewej strony lista dostępnych wtyczek.



Rysunek 17. Główne okno aplikacji z wtyczką APDU

Graficzny interfejs użytkownika korzysta z biblioteki WeifenLuo DockPanel Suite, która jest udostępniana bezpłatnie wraz z kodem źródłowym [28]. Biblioteka ta umożliwia zarządzanie oknami potomnymi, pozwala na dokowanie ich wewnątrz aplikacji oraz wyświetlanie jako niezależnych okien.

5.1.2. Interfejs głównej aplikacji

Główna aplikacja implementuje interfejs IMainApplication. Ułatwia to komunikację wtyczki z główną aplikacją, przesyłanie poleceń APDU do karty oraz komunikację z innymi wtyczkami. Interfejs głównej aplikacji wygląda następująco:

```
public interface IMainApplication
{
    /// <summary>
    /// Przekazanie do głównej aplikacji żądania wykonania polecenia
    /// </summary>
    /// <param name="command">Żądane polecenie</param>
    /// <param name="parameters">Parametry wywoływanego polecenia</param>
    /// <returns>Wynik wykonania polecenia</returns>
    Hashtable Exec(string command, Hashtable parameters);

    /// <summary>
    /// Lista załadowanych wtyczek
    /// </summary>
    List<IPlugin> Plugins { get; }

    /// <summary>
    /// Terminal kart elektronicznych
    /// </summary>
    IEncoder Encoder { get; }

    /// <summary>
    /// Zarządca terminali kart elektronicznych
    /// </summary>
    CardTerminalManager TerminalManager { get; }
}
```

Metoda *Exec* jest przeznaczona do wywoływania poleceń udostępnionych w głównej aplikacji, które nie zostały przewidziane podczas projektowania interfejsu. Dzięki temu nie ma konieczności przebudowy interfejsu, gdy zostanie wprowadzona zmiana w głównej aplikacji.

Pole *Plugins* zawiera listę załadowanych wtyczek, do których można się odwoływać i korzystać z ich funkcji.

Pole *Encoder* służy do komunikacji z terminalem kart elektronicznych i pozwala na przesyłanie poleceń APDU oraz odbiór odpowiedzi.

Pole *TerminalManager* musi być używane z rozwagą. Jest to zarządca terminali kart elektronicznych. Pole jest używane przez wtyczkę Terminal Listing, która wyświetla listę czytników. Do tego pola można przypisać zdarzenia włożenia i wyjęcia karty do/z czytnika i obsługa zdarzeń zostanie wykonana przez odpowiednią wtyczkę. Ważne, aby podczas wyłączania wtyczki obsługę zdarzeń przekazać do głównej aplikacji.

5.1.3. Obsługa terminali kart elektronicznych

Czytniki kart elektronicznych są obsługiwane przez bibliotekę Subsembly SmartCard Express udostępnianą bezpłatnie przez producenta [29]. Obsługa kart jest oparta na zdarzeniach włożenia i wyjęcia karty do/z czytnika. Włożenie karty powoduje odczyt numeru ATR karty i jej rozpoznanie

na podstawie danych z bazy danych w postaci pliku XML (zawiera on listę numerów ATR oraz rodzaje kart przypisane do tych numerów, listę można w łatwy sposób rozbudować).

5.2. Implementacja wtyczek

Wtyczki do programu SmartCard Suite 2.0 implementują interfejs *IPlugin*. Ujednolica to komunikację pomiędzy wtyczką i główną aplikacją a także ułatwia komunikację pomiędzy wtyczkami. Interfejs wtyczki został przedstawiony poniżej:

```
public interface IPlugin
{
    string PluginName { get; }
    string Description { get; }
    string Author { get; }
    string Version { get; }
    string Date { get; }

    DockContent PluginForm { get; }

    IMainApplication application { get; }

    /// <summary>
    /// Wywołuje wykonanie dowolnego polecenia z pluginu
    /// </summary>
    /// <param name="command">Nazwa polecenia</param>
    /// <param name="parameters">Parametry wejściowe</param>
    /// <returns>Wynik wywołania</returns>
    Hashtable Exec(string command, Hashtable parameters);

    /// <summary>
    /// Metoda wywoływana podczas uruchamiania pluginu
    /// </summary>
    /// <param name="initializationParameters">Parametry inicjalizacyjne</param>
    void PluginLoad(Hashtable initializationParameters);

    /// <summary>
    /// Metoda wywoływana podczas wyłączania pluginu
    /// </summary>
    /// <param name="parameters">Parametry końcowe</param>
    void PluginUnLoad(Hashtable parameters);

    /// <summary>
    /// Tworzy formę (lub upewnia się, że forma istnieje)
    /// </summary>
    /// <param name="force">Czy wymusić utworzenie nowej formy</param>
    void CreatePluginForm(bool force = false);
}
```

Znaczenia poszczególnych pól są następujące:

- pole *PluginName* oznacza nazwę wtyczki wyświetlaną na liście wtyczek,
- pole *Description* zawiera opis wtyczki, który pojawia się w oknie z informacjami o programie,
- pole *Author* zawiera dane autora,
- pole *Version* zawiera numer wersji wtyczki,
- pole *Date* zawiera datę przygotowania wtyczki.

Zaleca się, aby wszystkie pola były uzupełnione. Pole *PluginForm* typu *DockContent* dziedziczy z klasy *Form* i stanowi okno wtyczki, które zostanie wyświetlone po uruchomieniu wtyczki. Pole *application* jest odwołaniem do głównej aplikacji i umożliwia komunikację z aplikacją oraz z innymi wtyczkami.

Metoda *Exec* ma znaczenie takie, jak metoda o tej samej nazwie w interfejsie *IMainApplication*. Umożliwia ona uruchamianie poleceń nieprzewidzianych podczas przygotowania interfejsu.

Metoda *PluginLoad* wywoływana jest podczas uruchamiania wtyczki. W niej mogą znaleźć się np. polecenia przechwytujące zdarzenie włożenia karty.

Metoda *PluginUnLoad* jest wywoływana podczas kończenia pracy wtyczki – przechwycone wcześniej zdarzenie musi zostać zwolnione.

Metoda *CreatePluginForm* tworzy nowe okno wtyczki, o ile wcześniej nie istniało, w przeciwnym wypadku nic nie robi, w związku z czym automatycznie nie jest otwierane drugie okno dla jednej wtyczki. Jeśli jest konieczne otwarcie drugiego okna należy podać parametr o wartości *True*.

5.2.1. Tworzenie wtyczek

Wraz z aplikacją jest publikowany plik *SmartCard Suite 2.0 Plugin.zip*, który zawiera szablon do tworzenia nowych wtyczek. Aby z niego korzystać należy skopiować plik do katalogu *%HOMEPATH%\Documents\Visual Studio 2010\Templates\ProjectTemplates*. Działanie szablonu zostało sprawdzone w oprogramowaniu Microsoft Visual Studio 2010 Ultimate na systemie Windows 7 64 bit.

6. PODSUMOWANIE

Celem niniejszej pracy było przygotowanie uniwersalnej aplikacji obsługującej karty elektroniczne wraz z podstawowym zestawem wtyczek oraz przygotowanie scenariuszy zajęć laboratoryjnych. Obsługa wielu rodzajów kart elektronicznych w jednej aplikacji jest trudna, ponieważ różne karty wymagają odmiennego sposobu komunikacji. Cel pracy został osiągnięty. Stworzona została aplikacja obsługująca kilka rodzajów kart elektronicznych. Pozwala ona na zarządzanie zainstalowanymi w systemie terminalami kart elektronicznych. Obsługa różnego rodzaju kart została osiągnięta z wykorzystaniem systemu wtyczek, dzięki czemu oprogramowanie charakteryzuje się łatwością rozbudowy. Przygotowane wtyczki obejmują podstawowe możliwości kart elektronicznych takich jak Elektroniczna Legitymacja Studencka, karty SIM oraz karty z pamięcią MIFARE.. Przedstawione schematy zajęć laboratoryjnych obejmują większość najpopularniejszych rodzajów kart elektronicznych.

Wdrożenie oprogramowania SmartCard Suite 2.0 planowane jest przed rozpoczęciem roku akademickiego 2012/2013. Do tego czasu należy przygotować wtyczki obsługujące karty używane na poszczególnych zajęciach.

Autor uznał za bezcelowe tworzenie wtyczek zastępujących środowiska programistyczne: Microsoft Visual Studio dla kart .Net, Eclipse dla kart JavaCard oraz ZeitControl dla kart BasicCard. Środowiska te składają się z zestawów wielu narzędzi wspomagających pracę i ciągle są rozwijane. By móc zastąpić środowisko programistyczne należałoby stworzyć wiele wtyczek.

Wskazówki dalszego rozwoju

Rozwój zajęć laboratoryjnych może polegać na dodaniu nowego rodzaju karty do programu zajęć. Mogłyby to być np. karty cyfrowej telewizji satelitarnej i naziemnej. W celu równoległego rozwoju aplikacji należy przygotować odpowiednią wtyczkę obsługującą nowy rodzaj kart elektronicznych. W tym celu przygotowano szablon wtyczki dla Visual Studio 2010 w postaci stanowi archiwum zip znajdującego się na dołączonej płycie.

W programie zajęć laboratoryjnych znajdują się karty EMV. Niestety ogólnodostępne oprogramowanie odczytujące dane z kart tego typu zawiera wiele błędów. Główni dostawcy kart EMV – Visa i MasterCard, nie publikują dokumentacji, co utrudnia tworzenie oprogramowania. Zaleca się podjęcie prób uzyskania odpowiedniej dokumentacji i przygotowania wtyczki obsługującej karty EMV. Ponieważ normy dotyczące kart EMV nie określają precyzyjnie oznaczenia plików, alternatywną metodą ich poznania jest brutalny atak (ang. *Brute – force*) w celu odkrycia nazw plików znajdujących się w pamięci karty i próba odczytu znalezionych plików.

LITERATURA

Książki, publikacje

[1] Bartecki M., Elektroniczna legitymacja studencka – implementacja podstawowych czynności obsługi w dziekanacie, Politechnika Poznańska, 2008.

[2] Bellamy E., „W roku 2000.", tł. J. K. Potocki, wyd. „GŁOSU", Warszawa, 1890.

[3] Gosławski M., Materiały dydaktyczne do laboratorium z przedmiotu Programowanie kart elektronicznych, Politechnika Poznańska (materiały niepublikowane).

[4] Mika M., Wykłady z przedmiotu Programowanie kart elektronicznych, Politechnika Poznańska, 2011.

[5] Nazimek P., Inżynieria programowania kart inteligentnych, Politechnika Warszawska, 2005.

[6] Rankl W., Effing W., „Smart Card Handbook", wydanie trzecie, wyd. John Wiley & Sons, Munich, 2002.

[7] Tomczak J., Aplikacja biblioteczna jElib, Politechnia Poznańska, 2009, praca magisterska.

Normy, specyfikacje, akty prawne

[8] EMV, Integrated Circuit Card Specifications for Payment Systems, Book 1, Version 4.3.

[9] EMV, Integrated Circuit Card Specifications for Payment Systems, Book 2, Version 4.3.

[10] EMV, Integrated Circuit Card Specifications for Payment Systems, Book 3, Version 4.3.

[11] EMV, Integrated Circuit Card Specifications for Payment Systems, Book 4, Version 4.3.

[12] ETSI TS 100 977 v8.14.0: 2007, Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module – Mobile Equipment (SIM-ME) Interface (3GPP TS 11.11 version 8.14.0 Release 1999).

[13] ISO/IEC 14443-4: 2008, Karty identyfikacyjne – Zbliżeniowe układy scalone – Karty zbliżeniowe – Część 4: Protokół transmisji danych.

[14] ISO/IEC 7810: 2003, Identification Cards – Physical characteristic.

[15] ISO/IEC 7816-2: 2007, Information technology - Identification cards - Integrated circuit cards - Part 2: Cards with contacts - Dimensions and location of the contacts.

[16] ISO/IEC 7816-3: 1997, Information technology – Identification cards – Integrated circuit(s) cards with contacts – Part 3: Electronic signals and transmission protocols.

[17] ISO/IEC 7816-4: 1995, Information technology – Identification cards – Integrated circuit(s) cards with contacts – Part 4: Interindustry commands for interchange.

[18] MF1ICS50: 2010, Functional specification, Rev. 5.6.

[19] Rozporządzenie Ministra Nauki i Szkolnictwa Wyższego z dnia 2 listopada 2006 r. w sprawie dokumentacji przebiegu studiów.

[20] Java Card Classic Platform Specification, wersja 3.0.4.

Strony internetowe

[21] Oprogramowanie Edsi Easy Reader, http://edsi-smartcards.com, dostęp 14.05.2012.

[22] Oprogramowanie Omnikey Diagnostic Tool, http://www.hidglobal.com, dostęp 14.05.2012.

[23] Oprogramowanie ASN.1 Editor, http://lipingshare.com/Asn1Editor/, dostęp 14.05.2012.

[24] Oprogramowanie Application Loader i AuthentIC Webpack, http://www.oberthur.com, dostęp 14.05.2012.

[25] Oprogramowanie ZeitControl BasicCard, http://www.zeitcontrol.de/index_e.htm, dostęp 14.05.2012.

[26] Oprogramowanie Smart Card Shell, http://www.openscdp.org/scsh3/, dostęp 14.05.2012.

[27] Oprogramowanie TrueCrypt, http://www.truecrypt.org/, dostęp 14.05.2012.

[28] Biblioteka WeifenLuo Dock panel suite, http://dockpanelsuite.sourceforge.net/, dostęp 5.02.2012.

[29] Biblioteka Subsembly SmartCard Express, http://www.smartcard-api.com/index.shtml, dostęp 18.02.2012.

[30] Zdjęcie z aukcji, http://www.etsy.com/listing/51719675/vintage-charge-plate-hutzler-brothers, dostęp 20.03.2012.

[31] Zdjęcie styków kart elektronicznych, http://en.wikipedia.org/wiki/Smart_card, dostęp 20.03.2012.

[32] Dokumentacja kart .NET wraz z wymaganymi sterownikami, http://www.gemalto.com/products/dotnet_bio/resources/development.html, dostęp 12.04.2012.

DODATEK A – LISTA UŻYWANYCH SKRÓTÓW

APDU	Jednostka danych protokołu aplikacji (ang. *Application Protocol Data Unit*).
ASN.1	Abstrakcyjna notacja składniowa 1 (ang. *Abstract Syntax Notation One*).
ATR	Odpowiedź na zerowanie (ang. *Answer To Reset*).
ELS	Elektroniczna Legitymacja Studenta.
IMSI	Numer identyfikacyjny ITU-T przydzielany przez dostawcę dla telefonów niestacjonarnych (ang. *International Mobile Subscriber Identity*).
MCPLS (MCP)	Międzyuczelniane Centrum Personalizacji Legitymacji Studenckich.
MDI	Interfejs wielodokumentowy (ang. *Multi Document Interface*).
MIFARE	Bezstykowy standard kart opracowany przez firmę Philips w 1994 roku.
PEKA	Poznańska Elektroniczna Karta Aglomeracyjna.
PIN	Kod wymagany do korzystania z karty SIM (używany również na innych typach kart elektronicznych pod pojęciem CHV, osobisty numer identyfikacyjny, ang. *Personal Identification Number*).
PKCS#11	System kryptograficzny klucza publicznego (ang. *Public Key CryptoSystems*).
PUK	Kod służący do odblokowania kodu PIN po jego kilkukrotnym błędnym wprowadzeniu (ang. *Personal Unlock Key*).
SIM	Moduł identyfikacji abonenta (ang. *Subscriber Identity Module*).
TLV	Format zapisu danych typ - długość - wartość (ang. *Type-Length-Value*). W dokumentacjach do niektórych kart jest podane jako tag - długość - wartość (ang. *Tag-Length-Value*).
TPDU	Protokół transmisji APDU (ang. *Transmission Protocol Data Unit*).
UID	Unikalny identyfikator zapisany na kartach elektronicznych z interfejsem bezstykowym w pamięci MIFARE (ang. *Unique IDentifier*).

DODATEK B – SPIS RYSUNKÓW I TABEL

Spis rysunków

Rysunek 1. Przykładowa karta Charge-Plate [30] 9
Rysunek 2. Wymiary karty formatu ID-1 [4] 11
Rysunek 3. Wymiary karty formatu ID-000 [4] 11
Rysunek 4. Porównanie formatów kart elektronicznych [4] 11
Rysunek 5. Wymiarowanie styków na karcie elektronicznej [4] 12
Rysunek 6. Różne kształty styków [31] 12
Rysunek 7. Przykładowa struktura plików w karcie elektronicznej [17] 15
Rysunek 8. Edsi Easy Reader - główne okno programu 23
Rysunek 9. Wygląd ekranu programu Omnikey Diagnostic Tool 23
Rysunek 10. Główne okno programu z otwartym przykładowym plikiem 24
Rysunek 11. Okno programu SELSTerminal 25
Rysunek 12. Ekran aplikacji ZeitControl Basic Card z wczytanym przykładowym programem 26
Rysunek 13. Główne okno programu Smart Card Shell 26
Rysunek 14. Okno programu True Crypt z listą tokenów zapisanych na karcie 27
Rysunek 15. Okno programu AuthentIC wyświetlające informacje o karcie 27
Rysunek 16. Architektura oprogramowania SmartCard Suite 2.0 43
Rysunek 17. Główne okno aplikacji z wtyczką APDU 45

Spis tabel

Tabela 1. Struktura polecenia APDU w protokole T=0 13
Tabela 2. Struktura odpowiedzi APDU w protokole T=0 13
Tabela 3. Struktura bloku TPDU w protokole T=1 14

DODATEK C – ZAWARTOŚĆ DVD

Na załączonej płycie DVD znajduje się niniejsza praca w formacie DOC i PDF a także przygotowane oprogramowanie – aplikacja SmartCard Suite 2.0 z podstawowym zestawem wtyczek oraz archiwum ZIP będące szablonem wtyczki.