

Laboratorium

Programowania Kart Elektronicznych

Programowanie JavaCard

Marek Goślawski

Programowanie JavaCard

- Przygotowanie do zajęć
 - dokumentacja JavaCard i GlobalPlatform
 - środowisko programistyczne
 - karta JavaCard
- Potrzebne wiadomości
 - język angielski w stopniu pozwalającym na czytanie dokumentacji technicznej
 - składnia poleceń i odpowiedzi APDU
 - podstawowa znajomość składni języka Java

Programowanie JavaCard

- GlobalPlatform
 - Runtime Environment
 - Card Manager
 - Command dispatch
 - Card content management
 - Security management
 - Security Domain
 - API
 - Card Content

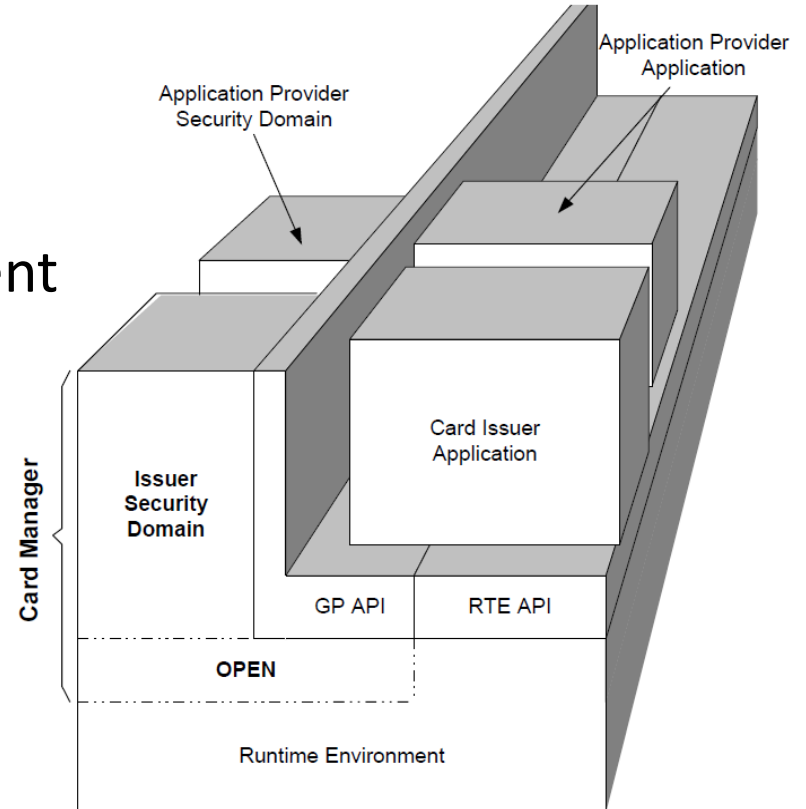
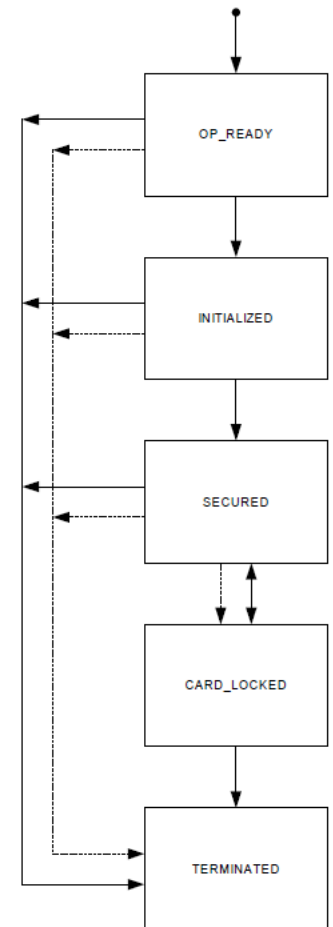


Figure 3-1: GlobalPlatform Card Architecture

Programowanie JavaCard

- Stany karty
 - OP_READY
 - INITIALIZED
 - SECURED
 - CARD_LOCKED
 - TERMINATED



Legend

Issuer Security Domain ———

Privileged Application - - - - -

Figure 5-1: Card Life Cycle State Transitions

Programowanie JavaCard

Command	OP_READY			INITIALIZED			SECURED			CARD_LOCKED			TERMINATED		
	ISD	DM SD	SD	ISD	DM SD	SD	ISD	DM SD	SD	ISD	DM SD	SD	ISD	DM SD	SD
DELETE Executable Load File															
DELETE Executable Load File and related Application(s)															
DELETE Application	✓			✓			✓								
DELETE Key															
GET DATA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓		
GET STATUS	✓			✓			✓			✓					
INSTALL [for load]															
INSTALL [for install] (*)	✓	✓		✓	✓		✓	✓							
INSTALL [for make selectable] (*)	✓	✓		✓	✓		✓	✓							
INSTALL [for extradition]															
INSTALL [for personalization]															
LOAD															
PUT KEY	✓			✓			✓								
SELECT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓					
SET STATUS	✓			✓			✓			✓					
STORE DATA	✓			✓			✓								

Table 9-1: Authorized GlobalPlatform Commands per Card Life Cycle State

ISD: Issuer Security Domain.

DM SD: Application Provider Security Domain with Delegated Management privilege.

SD: Application Provider Security Domain without Delegated Management privilege.

Programowanie JavaCard

- Ładowanie apletu do pamięci karty
 - SELECT
 - LOAD
 - INSTALL

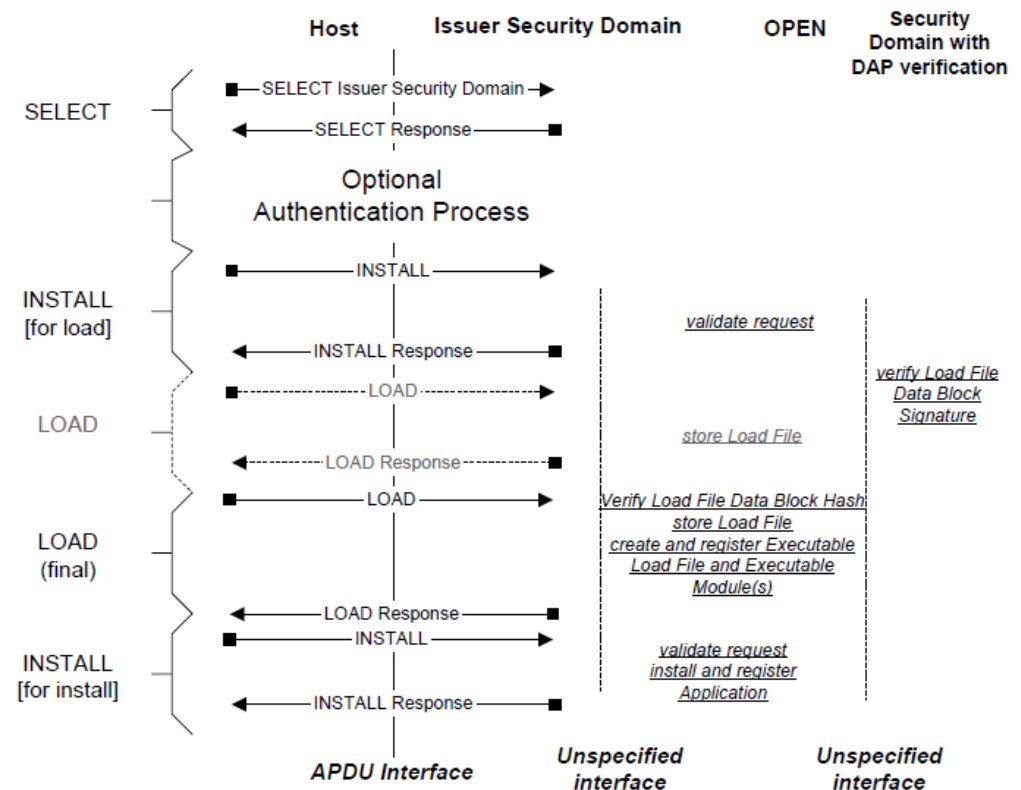
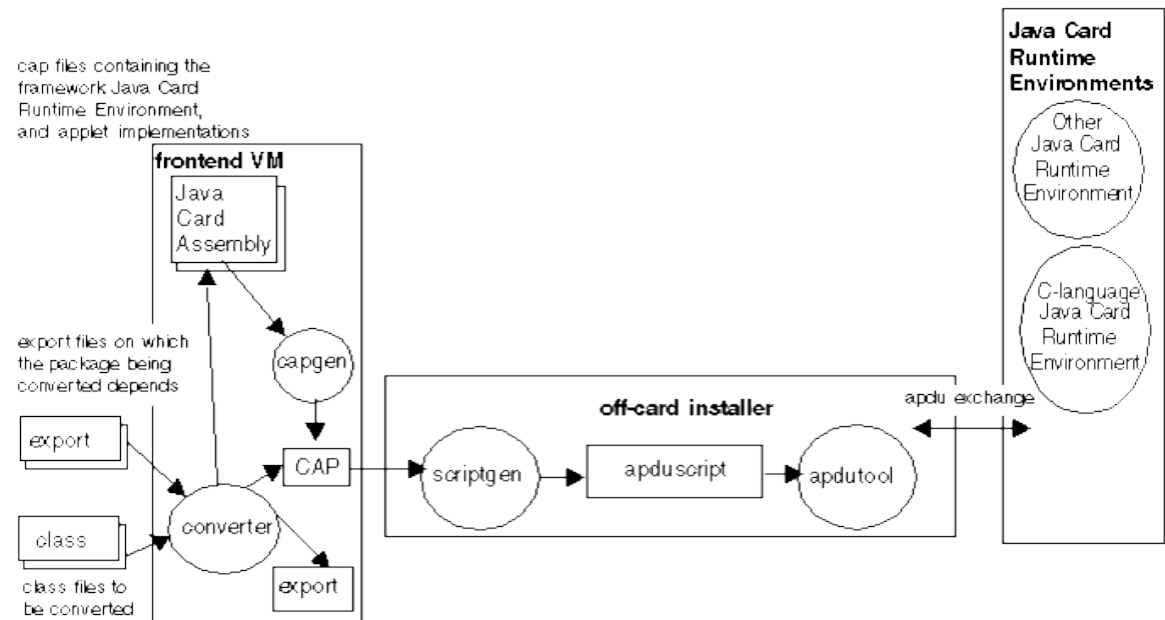


Figure 6-3: Load and Installation Flow Diagram

Programowanie JavaCard

- Oprogramowanie
 - środowisko JavaCard 2.2.1
 - Development Kit User's Guide (cJDK_Users_Guide.pdf)
 - JavaCard 2.2.1 → Java 1.4



Programowanie JavaCard

- Rodzaje plików
 - kod źródłowy aplikacji (*.java)
 - skompilowany kod aplikacji (*.class)
 - skonwertowane pliki aplikacji (*.jar, *.jca, ***.cap**)
 - JCA (java card assembly) – „human-readable” 😊
 - CAP (converted applet)
 - pliki z informacjami o klasach (*.exp)
 - pliki skryptów (*.scr)

Programowanie JavaCard

- Elementy języka Java

- typy danych

- boolean, char, byte, short, int, long, float, double, void
 - obiekty!

- deklaracje danych

```
char c = 'x'
```

```
Character ch = new Character('x');
```

```
String s = new String("a string");
```

Programowanie JavaCard

- Elementy języka Java

- wyrażenia

```
x + y - 2/2 + z;  
x * (y - 2)/(2 + z);
```

- przypisania

```
int x = 1, y = 2, z = 3;  
int a = x + y - 2/2 + z;  
int b = x + (y - 2)/(2 + z);
```

- polecenia kontrolujące działanie programu

```
if(Boolean-expression) <statement> else <statement>  
while(Boolean-expression) <statement>  
do <statement> while(Boolean-expression)  
for(initialization; Boolean-expression; step) <statement>  
break  
continue
```

Programowanie JavaCard

- Elementy języka Java
 - klasy, pola, metody

```
public class Documentation1 {  
    public int i;  
    public void f() {}  
    public int g() {  
        return 0;  
    }  
    public static void main(String[] args) {  
        System.out.println(new Date());  
    }  
}
```

Programowanie JavaCard

- Przykłady

- program karty

```
import javacard.framework.*;
public class HelloWorld extends Applet {

    public static void install(
        byte[] bArray, short bOffset, byte bLength) {}

    public void process(APDU apdu) {
        byte buffer[] = apdu.getBuffer();
        apdu.setOutgoing();
        apdu.setOutgoingLength( (short) (echoOffset + 5) );
        apdu.sendBytes( (short)0, (short) 5);
        apdu.sendBytesLong( echoBytes, (short) 0, echoOffset );
    }
}
```

Programowanie JavaCard

- Zadania
 - program „Hello <student>”
 - symulator (program i logi, 5 poleceń APDU)
 - karta (logi komunikacji z kartą, 5 poleceń APDU)
- przesłanie na adres:
 - marek.goslowski@put.poznan.pl

Zadanie 1: „Hello <student>” – symulator

Zadanie 2: „Hello <student>” – karta

wysłać na adres:

marek.goslowski@put.poznan.pl

Programowanie JavaCard

- Uruchomienie środowiska karty
 - kompilacja kodu
 - `javac.exe` (z j2sdk)
 - należy dobrać wersję java do wersji javacard
 - wymagane biblioteki z javacard
 - wejście `*.java`, wyjście `*.class`
 - konwersja plików wynikowych
 - `converter.bat` (z jcsdk2.2)
 - wejście `*.class`, wyjście `*.jca`, `*.exp`, `*.cap`

Zadanie 1: „Hello <student>” – symulator

Zadanie 2: „Hello <student>” – karta

wysłać na adres:

marek.goslowski@put.poznan.pl

Programowanie JavaCard

- Uruchomienie środowiska karty
 - generowanie skryptów
 - `scriptgen.bat` (z `jcsdk2.2`)
 - wejście `*.cap`, wyjście `*.scr`
 - C-language Java Card Runtime Environment
 - `ceref.exe` (z `jcsdk2.2`)
 - wejście `*.scr`

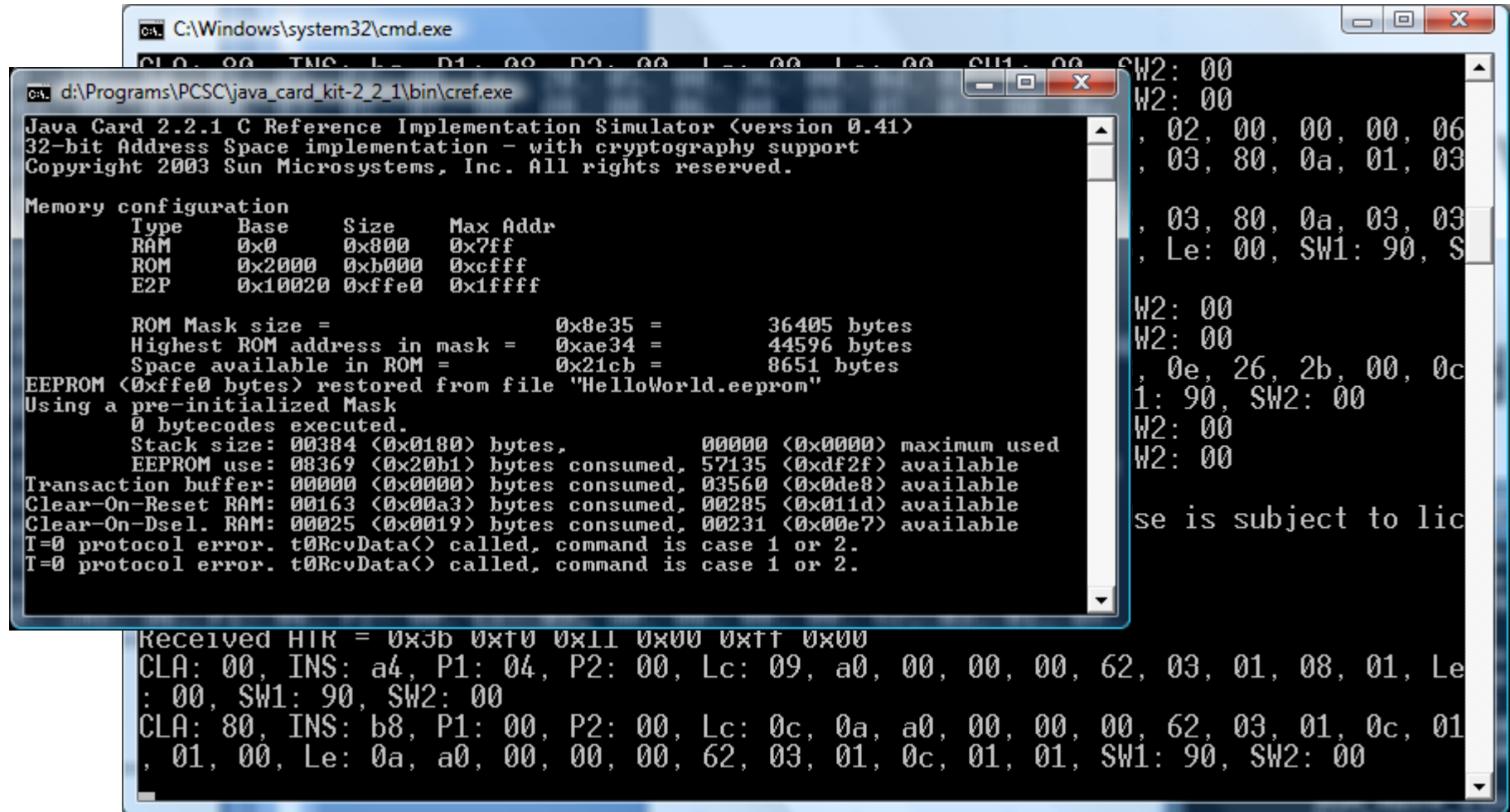
Zadanie 1: „Hello <student>” – symulator

Zadanie 2: „Hello <student>” – karta

wysłać na adres:

marek.goslowski@put.poznan.pl

Programowanie JavaCard



The image shows two overlapping windows. The top window is a command prompt titled 'C:\Windows\system32\cmd.exe' showing the execution of 'd:\Programs\PCSC\java_card_kit-2_2_1\bin\cref.exe'. The bottom window is the 'Java Card 2.2.1 C Reference Implementation Simulator (version 0.41)'.

Java Card 2.2.1 C Reference Implementation Simulator (version 0.41)
32-bit Address Space implementation - with cryptography support
Copyright 2003 Sun Microsystems, Inc. All rights reserved.

Memory configuration

Type	Base	Size	Max Addr
RAM	0x0	0x800	0x7ff
ROM	0x2000	0xb000	0xcfff
E2P	0x10020	0xffe0	0x1ffff

ROM Mask size = 0x8e35 = 36405 bytes
Highest ROM address in mask = 0xae34 = 44596 bytes
Space available in ROM = 0x21cb = 8651 bytes

EEPROM (0xffe0 bytes) restored from file "HelloWorld.eeprom"
Using a pre-initialized Mask
0 bytecodes executed.

Stack size: 00384 (0x0180) bytes, 00000 (0x0000) maximum used
EEPROM use: 08369 (0x20b1) bytes consumed, 57135 (0xdf2f) available
Transaction buffer: 00000 (0x0000) bytes consumed, 03560 (0x0de8) available
Clear-On-Reset RAM: 00163 (0x00a3) bytes consumed, 00285 (0x011d) available
Clear-On-Dsel. RAM: 00025 (0x0019) bytes consumed, 00231 (0x00e7) available

T=0 protocol error. t0RcvData() called, command is case 1 or 2.
T=0 protocol error. t0RcvData() called, command is case 1 or 2.

Received HIR = 0x3b 0xt0 0x11 0x00 0xtt 0x00
CLA: 00, INS: a4, P1: 04, P2: 00, Lc: 09, a0, 00, 00, 00, 62, 03, 01, 08, 01, Le: 00, SW1: 90, SW2: 00
CLA: 80, INS: b8, P1: 00, P2: 00, Lc: 0c, 0a, a0, 00, 00, 00, 62, 03, 01, 0c, 01, 01, 00, Le: 0a, a0, 00, 00, 00, 62, 03, 01, 0c, 01, 01, SW1: 90, SW2: 00

Zadanie 1: „Hello <student>” – symulator

Zadanie 2: „Hello <student>” – karta

wysłać na adres:

marek.goslowski@put.poznan.pl

Programowanie JavaCard

Java Card 2.2.1 APDU Tool, Version 1.3

Copyright 2003 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.

Opening connection to localhost on port 9025.

Connected.

Received ATR = 0x3b 0xf0 0x11 0x00 0xff 0x00

CLA: 00, INS: a4, P1: 04, P2: 00, Lc: 0a, a0, 00, 00, 00, 62, 03, 01, 0c, 01, 01,

Le: 0f, 00, a4, 04, 00, 0a, a0, 00, 00, 00, 62, 03, 01, 0c, 01, 01, SW1: 90, SW2: 00

CLA: 00, INS: 00, P1: 00, P2: 00, Lc: 00, Le: 00, SW1: 6f, SW2: 00

CLA: 00, INS: 00, P1: 00, P2: 00, Lc: 05, a1, a2, a3, a4, a5,

Le: 0a, 00, 00, 00, 00, 05, a1, a2, a3, a4, a5, SW1: 90, SW2: 00

CLA: a0, INS: a1, P1: a2, P2: a3, Lc: 00, Le: 00, SW1: 6f, SW2: 00

CLA: a0, INS: a1, P1: a2, P2: a3, Lc: 05, b1, b2, b3, b4, b5,

Le: 0a, a0, a1, a2, a3, 05, b1, b2, b3, b4, b5, SW1: 90, SW2: 00

CLA: ff, INS: ff, P1: ff, P2: ff, Lc: 05, a1, a2, a3, a4, a5,

Le: 0a, ff, ff, ff, ff, 05, a1, a2, a3, a4, a5, SW1: 90, SW2: 00

Zadanie 1: „Hello <student>” – symulator

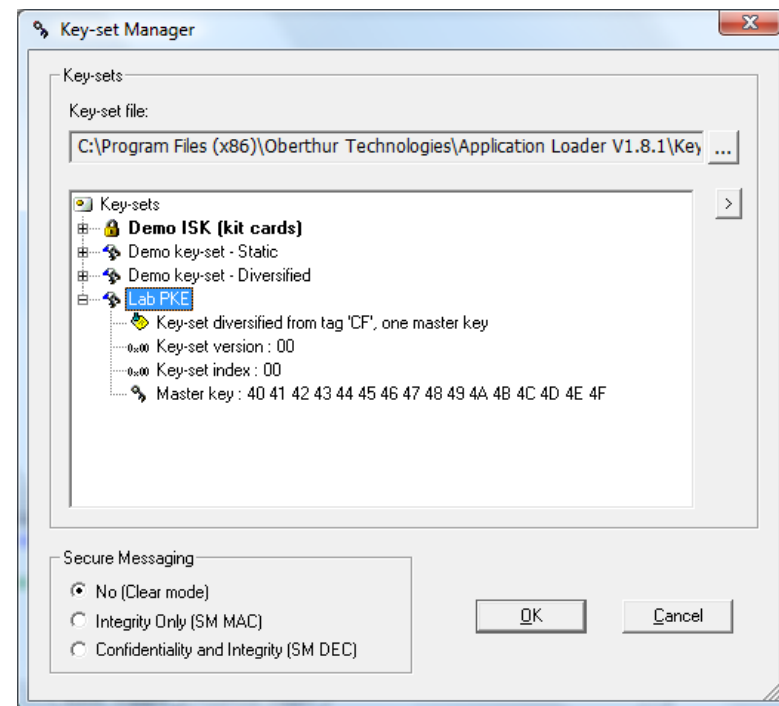
Zadanie 2: „Hello <student>” – karta

wysłać na adres:

marek.goslowski@put.poznan.pl

Programowanie JavaCard

- Instalacja na karcie
 - Application Loader
 - Klucze
 - jeden klucz „matka”
40 41 42 43 44 45 46 ... 4F
 - dywersyfikacja „from tag CF”
 - Card Manager AID:
A0 00 00 00 03 00 00 00



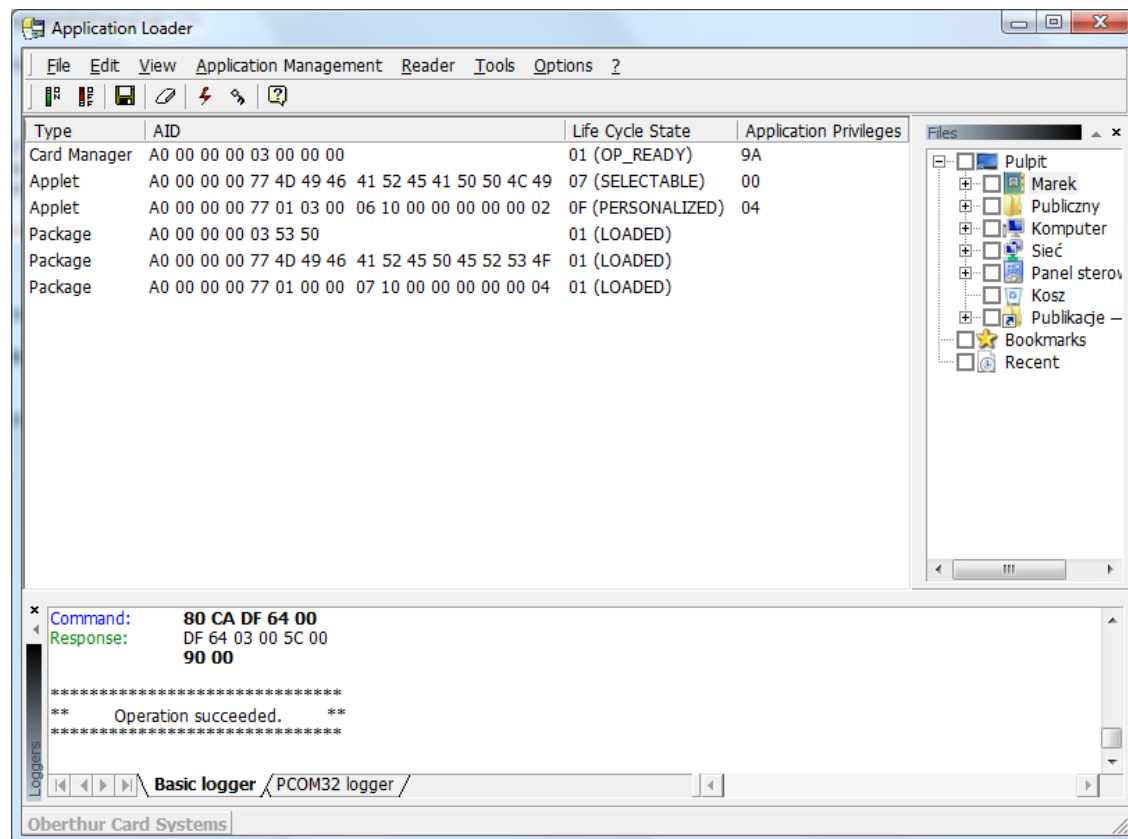
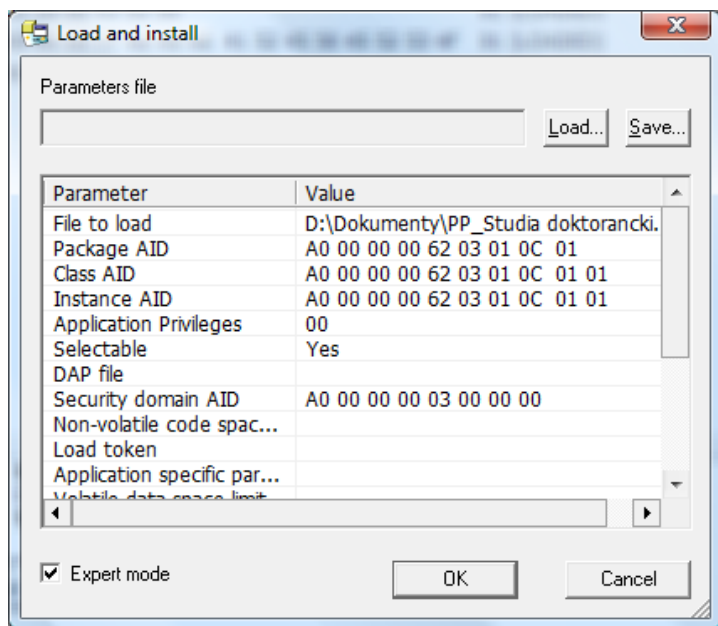
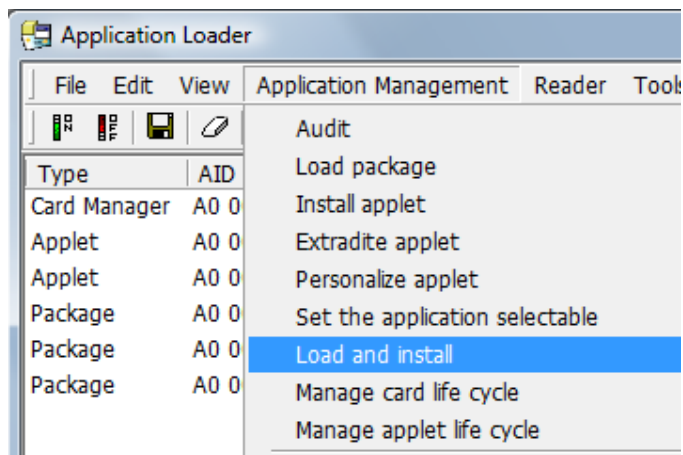
Zadanie 1: „Hello <student>” – symulator

Zadanie 2: „Hello <student>” – karta

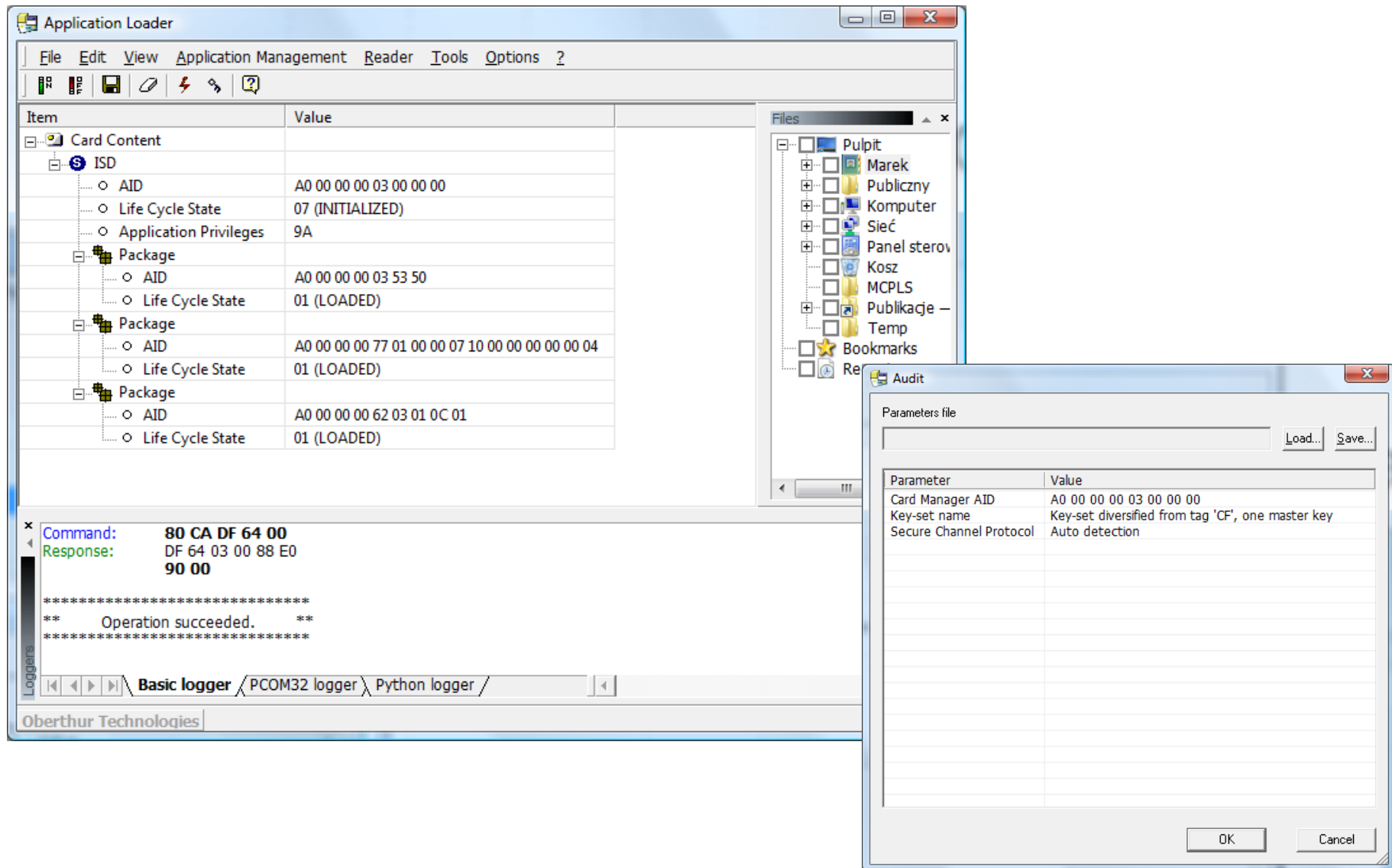
wysłać na adres:

marek.goslowski@put.poznan.pl

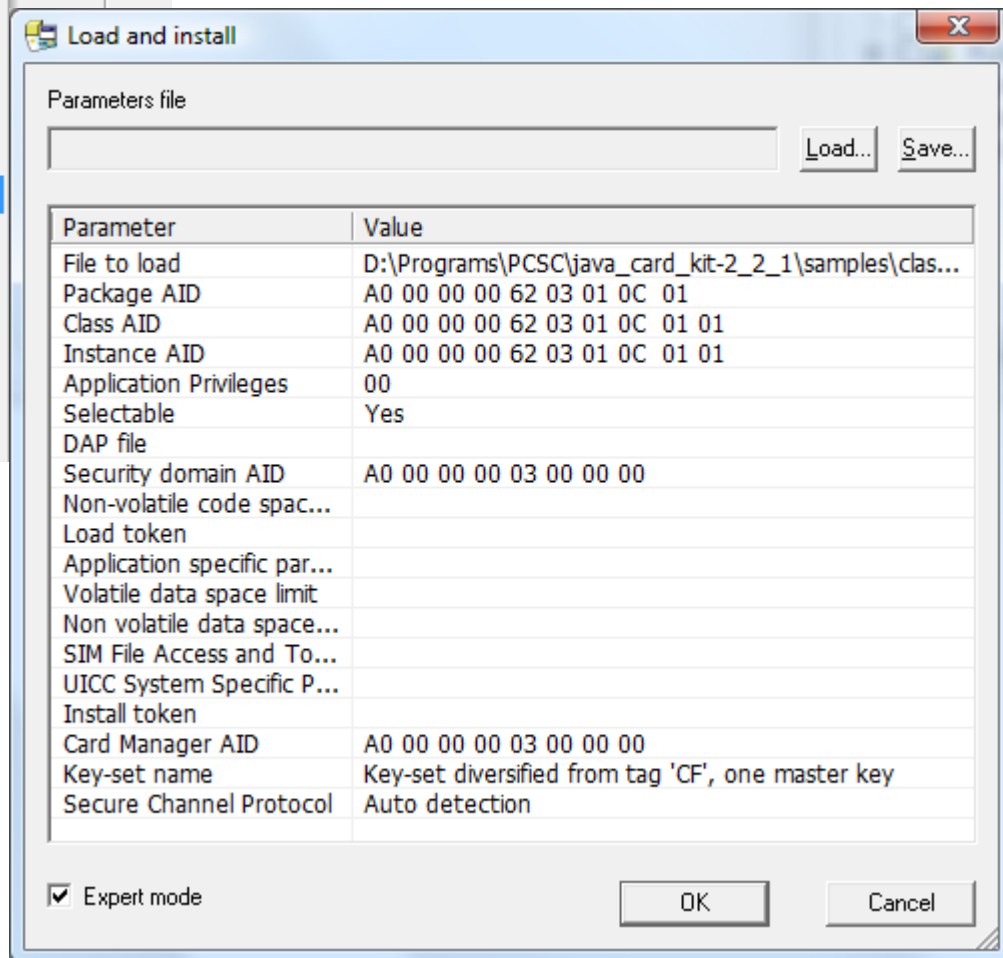
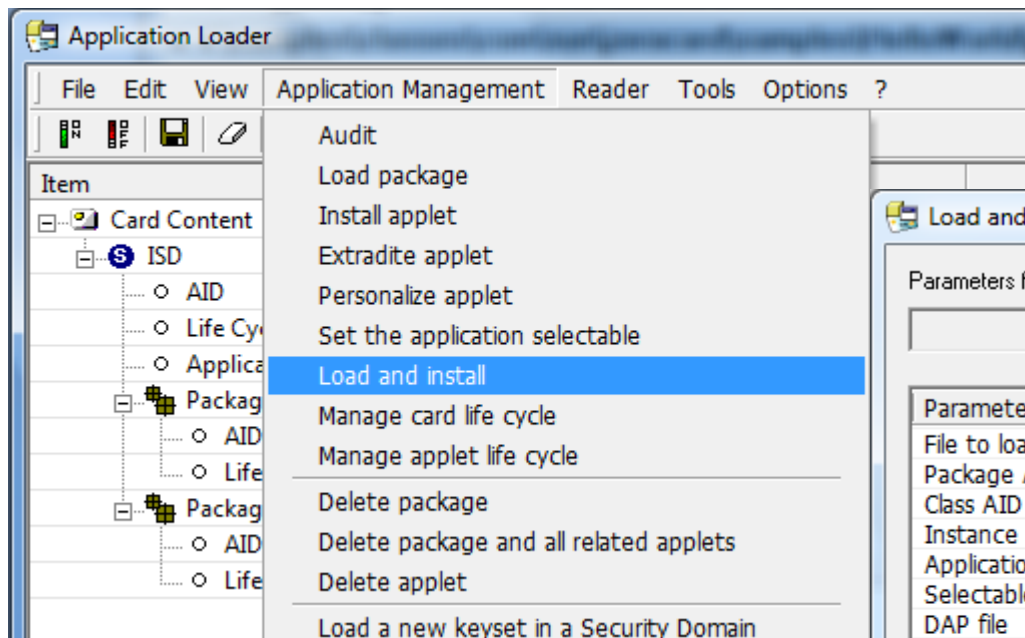
Programowanie JavaCard



Programowanie JavaCard



Programowanie JavaCard



Programowanie JavaCard

- GlobalPlatformPro tool
 - <https://github.com/martinpaljak/GlobalPlatformPro>

```
gp.exe -i
```

```
gp.exe -l -emv
```

```
gp.exe -emv --install helloworld.cap
```

```
gp.exe -emv --delete A000000006203010C01
```

```
PKG: A000000006203010C01 (LOADED)  
Version: 1.0  
Applet: A000000006203010C0101
```

- marek.goslowski@put.poznan.pl
- +48 61 665 3680
- +48 694 949 750
- pl. Marii Skłodowskiej-Curie 5 (Wilda)
Budynek B1 (Rektorat), pok. 405
- <http://mcp.poznan.pl/>