SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1654

VIŠENAMJENSKA PAMETNA KARTICA

Robert Đurin

Zagreb, srpanj 2007.

Zahvaljujem se,
mentoru doc. dr. sc. Marinu Golubu na stručnom vodstvu, te svojim roditeljima koji su me podupirali kroz školovanje

Sažetak

Ovaj rad bavi se izgradnjom programskog sustava za pametnu karticu u Java Card tehnologiji te izgradnjom programskog sučelja za komuniciranje s *appletom* na pametnoj kartici. Implementiran je sustav javnih ključeva (eng. *Public Key Infrastructure, PKI*) koji generira privatne i javne ključeve te certifikate. Uz to je implementirana i digitalna knjižnica koja koristi sustav javnih ključeva. Svrha digitalne knjižnice je da sprema dokumente u digitalnom obliku i da ih na zahtjev korisnika izdaje. Autentificiranje u sustav digitalne knjižnice obavlja se X509 protokolom autentifikacije s tri poruke. Prije spremanja dokumenta u digitalnu knjižnicu vrši se digitalno potpisivanje dokumenta te digitalno potpisivanje narudžbe prije preuzimanja dokumenta iz digitalne knjižnice.

Abstract

This thesis deals with building a software system for smart card in Java Card technology and with building a programming interface for communication with applet on smart card. Public key infrastructure (PKI) is implemented and his purpose is to generate private and public keys and certificates. Beside that digital library system is implemented which uses public key infrastructure. Purpose of digital library is to save documents in digital form and to issue documents.on user request. Authentification to the system is managed by X509 protocol based on three messages. Before the document is saved he is digitaly signed and before issuing an order is digitaly signed.

Sadržaj

1. UVOD 1

2. OSNOVNO O PAMETNIM KARTICAMA 2

2.1. Što je to pametna kartica 2

2.2. Vrste pametnih kartica 2
2.2.1. Memorijske kartice 2
2.2.2. Kontaktne mikroprocesorske kartice 2
2.2.3. Bezkontaktne mikroprocesorske kartice 3
2.2.4. Kombinirane mikroprocesorske kartice 4

2.3. Norme 4
2.3.1. ISO-7816 norma 4
2.3.1.1. ISO-7816-1 5
2.3.1.2. ISO-7816-2 5
2.3.1.3. ISO-7816-3 6
2.3.1.4. ISO-7816-4 6
2.3.1.5. ISO-7816-5 6
2.3.1.6. ISO-7816-6 7
2.3.2. EMV norma 7
2.3.3. GSM norma 7

2.4. Primjena pametnih kartica 7
2.4.1. Elektroničko plaćanje 8
2.4.2. Računalna sigurnost 8
2.4.3. Telekomunikacije 8
2.4.4. Zdrastvena kartica 8
2.4.5. Transport 9

2.5. Komunikacijski model 9
2.5.1. Protokoli za komuniciranje 9
2.5.1.1. T=0 protokol 9
2.5.1.2. T=1 protokol 10

2.6. APDU (eng. *Application Protocol Data Units*) 12
2.6.1. APDU naredba 12
2.6.2. APDU odgovor 13

3. JAVA CARD TEHNOLOGIJA 15

3.1. Java Card stogovni stroj 15

3.2. Java Card izvršna okolina 16
3.2.1. Životni ciklus Java Card stogovnog stroja 17
3.2.2. Životni ciklus Java Card appleta 17

3.3. Java Card programsko sučelje 18

4. OPENCARD TEHNOLOGIJA 20

4.1. Arhitektura OpenCard tehnologije 20
4.1.1. Terminalski sloj 21
4.1.2. Servisni sloj 22

4.2. Rad s OpenCard tehnologijom 22

5. IMPLEMENTACIJA JAVA CARD APPLETA 24

5.1. Funkcionalnost appleta 24
5.1.1. Autentifikacija i autorizacija 24
5.1.2. Kriptografija 25
5.1.2.1 Algoritam provjere privatnog ključa 25
5.1.3. Pohrana podataka 26

5.2. Implementacija appleta 26
5.2.1. Autentifikacija i autorizacija 27
5.2.2. Kriptografski algoritmi 27
5.2.3. Pohrana podataka 28
5.2.3.1. Organizacija memorije za pohranu podataka 29
5.2.3.2. Pisanje, čitanje i brisanje objekta 31
5.2.3.3. Autorizacija nad objektima 32

6. IMPLEMENTACIJA TERMINALSKOG SUČELJA 34

6.1. Menadžeri terminalskog sučelja 34
6.1.1. SelectFileManager 35
6.1.2. InitializeCardManager 35
6.1.3. CardInformationManager 35
6.1.4. PinManager 35
6.1.5. PukManager 35
6.1.6. CryptoManager 35
6.1.7. CardObjectManager 36

6.2. Dodatni razredi i sučelja 36
6.2.1. AbstractSmartCardApplet 36
6.2.2. CardService 37

6.3. Aplikacija za administraciju pametne kartice 37

7. SUSTAV JAVNIH KLJUČEVA (PKI) 42

7.1. Arhitektura PKI sustava 42
7.1.1. Certifikacijski centar (CA) 42
7.1.2. Registracijski centar (RA) 43
7.1.3. Spremnik certifikata i opozvanih certifikata 43
7.1.4. Izdavač liste opozvanih certifikata 43
7.1.5. Krajnji entiteti ili korisnici 43

7.2. Implementacija PKI sustava 43
7.2.1. Pametna kartica u PKI sustavu 46
7.2.1. Baza podataka PKI sustava 48
7.2.3. Tehnologije korištene kod implementacije PKI sustava 49

8. DIGITALNA KNJIŽNICA U SUSTAVU JAVNIH KLJUČEVA 51

8.1. Sigurnosni zahtjevi digitalne knjižnice 51

8.2. Arhitektura digitalne knjižnice 51

8.3. Autentifikacija i autorizacija korisnika 52
8.3.1. X.509 autentifikacijski protokol s tri poruke 54
8.3.2. Implementacija autentifikacijskog protokola 55
8.3.3. Implementacija autorizacije 56

8.4. Rad s dokumentima 58

8.5. Baza podataka digitalne knjižnice 60

8.6. Zaključna razmatranja 61

9. ZAKLJUČAK 62

10. LITERATURA 63

1. Uvod

Razvoj poluvodičke elektronike omogućio je minijaturizaciju integriranih krugova, a ta minijaturizacija omogućila je izradu računalnih sustava na jednom integriranom krugu (eng. *chip*). Takvi minijaturni računalni sustavi iskorišteni su za izradu pametnih kartica (eng. *smart card*) čija je osnova to malo mikroračunalo ugrađeno na plastičnu karticu. Pametna kartica omogućuje spremanje korisničkih podataka, zaštitu spremljenih podataka od neovlaštenog pristupa, enkripciju i dekripciju poruka, te mogućnost korištenja različitih usluga jednom pametnom karticom dok joj njezina veličina omogućuje laku prenosivost.

U ovom diplomskom radu demonstrira se upotreba pametne kartice u sustavu javnih ključeva (eng. *Public Key Infrastructure* skraćeno PKI). Sustav javnih ključeva ima funkciju generiranja i spremanja ključeva i certifikata. Sustav javnih ključev koristi digitalna knjižnica koja omogućuje spremanje i preuzimanje dokumenata te digitalno potpisivanje dokumenata. U sustavu digitalne knjižnice pametna kartica se koristi za autentifikaciju korisnika u sutav te za digitalno potpisivanje.

Na početku rada ukratko se opisuje pametna kartica, norme pametne kartice te komunikacijski model. Nakon toga opisana je Java Card tehnologija koja omogućuje izradu programa (*appleta*) za pametnu karticu te njegovo izvršavanje na pametnoj kartici. Slijedi OpenCard tehnologija koja omogućuje komunikaciju s pametnom karticom. Nakon opisa tehnologija slijedi opis implementacije appleta za pametnu karticu i terminala za komunikaciju s pametnom karticom koji su se izradili u svrhu praktičnog dijela ovog diplomskog rada. Slijedi opis PKI sustava u kojem se koristi pametna kartica. Na kraju je opisana digitalna knjižnica na kojoj se demonstrira rad pametne kartice i PKI sustava.

2. Osnovno o pametnim karticama

2.1. Što je to pametna kartica

Pametnu karticu (eng. *Smart Card*) možemo opisati kao plastičnu karticu s umetnutim integriranim krugom koji sadrži mikroprocesor. Pametne kartice ne sadrže u sebi nikakav uređaj za napajanje već napajanje dobivaju od uređaja za prihvat kartica CAD (eng. *Card Acceptance Device*). Za razliku od kartica s magnetnom trakom, pametne kartice pružaju puno veći nivo sigurnosti za podatke koji su na kartici, a također se jedna kartica može koristiti za više različitih usluga. Jedna od najvažnijih funkcija pametnih kartica smatra se činjenica da podaci spremljeni na njih mogu biti zaštićeni od neovlaštenog pristupa i mijenjanja podataka, a također omogućuju kriptiranje i dekriptiranje podataka.

2.2. Vrste pametnih kartica

Prema karakteristikama kartice s umetnutim integriranim krugom možemo podijeliti na:

1. memorijske kartice
2. kontaktne mikroprocesorske kartice
3. bezkontaktne mikroprocesorske kartice
4. kombinirane mikroprocesorske kartice

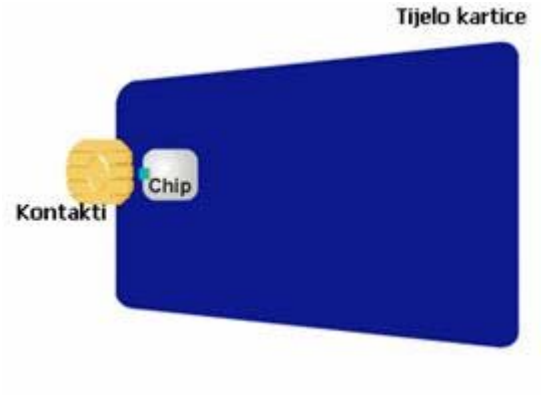
2.2.1. Memorijske kartice

Memorijske kartice su kartice koje na umetnutom integriranom krugu imaju samo memoriju i pristupnu logiku. Slično kao i kartice sa magnetnom trakom, memorijske mogu samo spremati podatke. U starijim memorijskim karticama nema ugrađene sigurne kontrolne logike. Zbog toga neovlašten pristup podacima na kartici ne može biti spriječen. Današnje memorijske kartice imaju u integrirani krug ugrađenu sigurnu logiku koja omogućuje da pristup sigurnoj zoni imaju samo autorizirani korisnici.

2.2.2. Kontaktne mikroprocesorske kartice

Kontaktne mikroprocesorske kartice imaju ugrađen mikroprocesor na umetnutom integriranom krugu, a prijenos podataka može biti samo onda kada je kartica umetnuta u uredaj za prihvat kartica CAD (eng. *Card Acceptance Device*) u svrhu čega kartica ima osam zlatnih kontakata na sebi kojima se električki spaja s CAD uređajem.

Vanjski izgled ovih kartica zasnovan je na ISO-7816-1 i ISO-7816-2 normama koje određuju dimenzije kartice, smještaj kontakata na kartici i njihov razmještaj. Struktura kontaktne pametne kartice prikazana je na slici 2.1.



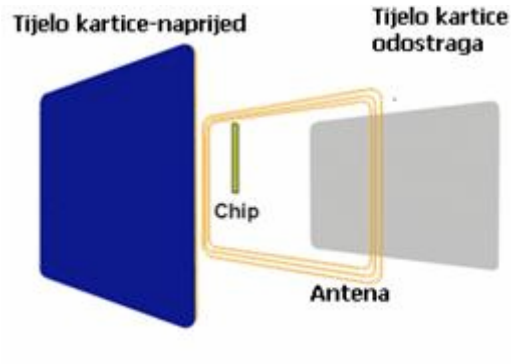
Slika 2.1. Kontaktna pametna kartica

Jedna od najvažnijih obilježja mikroprocesorskih pametnih kartica je sigurnost. Kontaktne mikroprocesorske pametne kartice su uglavnom prilagođene za sigurni prijenos podataka. Ako se korisnik ne autentificira uspješno, neće se moći pristupiti podacima spremljenim na kartici. Također, ako je kartica izgubljena, podaci spremljeni na kartici neće biti izloženi neovlaštenom pristupu ako su ti podaci prikladno spremljeni na kartici. Budući je pametna kartica malo računalo ona može sigurno obraditi unutarnje podatke te slati rezultat na izlazno sučelje.

2.2.3. Bezkontaktne mikroprocesorske kartice

Kontaktne pametne kartice nisu pogodne za sve tipove aplikacija, osobito kod masovnih transakcija i kada je potrebno da se podaci s kartice pročitaju u vrlo kratkom vremenu. Korištenjem elektromagnetskih valova bezkontaktne pametne kartice mogu slati podatke prema CAD uređaju s udaljenosti od nekoliko cm do 50 cm.

Kao i kontaktne kartice, ove također nemaju nikakvo unutarnje napajanje već koriste tehnologiju koja omogućava CAD uređajima da napaja i komunicira s karticom, bez fizičkog kontakta s karticom, preko elektromagnetskih valova. Struktura bezkontaktne kartice prikazana je na slici 2.2.



Slika 2.2. Bezkontaktna pametna kartica

Bezkontaktne pametne kartice su pogodne kod velikog broja pristupa kartici i prijenosa podataka, ali nisu pogodne kad je potreban prijenos velike količine podataka od kartice prema CAD uređaju. Također jedan od nedostataka bezkontaktnih kartica je da može doći do transakcije između CAD uređaja i kartice bez znanja korisnika.

2.2.4. Kombinirane mikroprocesorske kartice

Kontaktne i bezkontaktne kartice koriste dva različita protokola za komuniciranje s CAD uređajem. Obje kartice imaju svoje prednosti i nedostatke. Kontaktne pametne kartice imaju veći nivo sigurnosti, dok bezkontaktne kartice imaju bolje i jednostavnije sučelje za transakcijsko okruženje. U pokušaju da se korisnicima omoguće prednosti obiju kartica razvijena su dva načina:

1. Pametna kartica ima dva sučelja za komunikaciju, jedno je kontaktno, a drugo je bezkontaktno tako da kartica može komunicirati s CAD uređajem preko kontakata ili preko elektromagnetskih valova.
2. Kontaktna pametna kartica se umetne u poseban uređaj koji ima napajanje za karticu i antenu tako da može komunicirati preko elektromagnetskih valova s CAD uređajem.

2.3. Norme

Kroz povijest razvoja pametnih kartica uspostavljene su različite norme za rješavanje problema kompatibilnosti kartica različitih proizvođača. Prva norma bila je ISO-7816 koju je izdala međunarodna organizacija za norme (eng. *International Standard Organisation* - ISO) 1987. godine. Prije toga proizvođači kartica i CAD uređaja imali su svoje norme kartice i uređaje koji nisu bili kompatibilni s karticama i uređajima ostalih proizvođača. Sa ISO normom pametne kartice različitih proizvođača mogu komunicirati istim protokolom. Fizičke karakteristike i dimenzije su također jednake za sve kartice i one su također propisane normom, također je propisan položaj kontakata na kartici, značenje pojedinog kontakta te protokol i sadržaj poruka kod razmjene poruka. Ove norme osiguravaju da kartice proizvedene od jednog proizvođača mogu biti prihvaćene od CAD uređaja drugog proizvođača. Danas tri najvažnije norme su:

1. ISO-7816 norma
2. EMV (Europay, Mastercard, Visa) norma
3. GSM (Global Standard for Mobile Communications) norma

2.3.1. ISO-7816 norma

ISO-7816 je međunarodna norma za pametne kartice. ISO-7816 norma sastoji se od više dijelova od kojih su najbitniji:

1. ISO-7816-1
2. ISO-7816-2

3. ISO-7816-3
4. ISO-7816-4
5. ISO-7816-5
6. ISO-7816-6

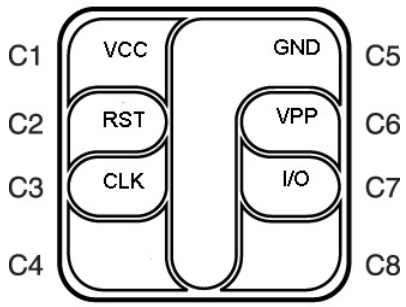
2.3.1.1. ISO-7816-1

Opisuje fizičke karakteristike pametnih kartica njezine dimenzije, te otpornost kartice na vanjske smetnje.

Dimenzije kartice po ovoj normi iznose 85.60mm × 53.98mm × 0.80mm. Ova norma propisuje i granice izlaganja kartice na razne vanjske utjecaje kao što su rendgenske zrake, UV svjetlo, elektromagnetska polja, elektrostatička polja, savijanje i rastezanje kartice.

2.3.1.2. ISO-7816-2

Određuje dimenzije i lokaciju kontakata na pametnoj kartici te funkciju i poziciju određenog kontakta. Kontaktna pločica na površini pametne kartice koja je spojena za izvodima integriranog kruga(čipa) umetnutog u pametnu karticu ima osam kontakata koji su označeni od C1 do C8. Međutim nije svih osam kontakata povezano s integriranim krugom i u današnje vrijeme ti kontakti još nemaju svoju specijalnu funkciju već su ti kontakti rezerviarni za buduću upotrebu. Na slici 2.3. prikazan je raspored kontakata, a u tablici 2.1. oznake i opis pojedinog kontakta.



Slika 2.3. Raspored kontakata pametne kartice

Kontakt	Oznaka	Opis
C1	Vcc	Kontakt za napajanje integriranog kruga – obično se kreće od +3V do +5V
C2	RST	Kontakt za reset preko kojeg se pokreće reset protokol
C3	CLK	Kontakt preko kojeg dovodimo takt signal integriranom krugu

C4	RFU	Rezervirano za buduću upotrebu (Reserved for Future Use)
C5	GND	Kontakt električne mase
C6	Vpp	Napajanje koje se u prošlosti koristilo za programiranje EEPROM memorije
C7	I/O	Ulazno/izlazni port, veza kartice s vanjskim svijetom, omogućuje half-duplex komunikaciju između čitača i CAD uređaja
C8	RFU	Rezervirano za buduću upotrebu (Reserved for Future Use)

Tablica 2.1. Oznake i opis kontakata pametne kartice

2.3.1.3. ISO-7816-3

Opisuje električne signale i protokole prenošenja signala kod pametnih kartica. Većina ove norme važna je za proizvođače CAD uređaja i za programere koji žele uspostaviti komunikaciju s pametnom karticom na vrlo niskom nivou, signalnom nivou. Ovom normom deklariraju se razine napona, jačine struja te perioda trajanja signala, te protokoli (T=0, T=1) komuniciranja između CAD uređaja i kartice.

2.3.1.4. ISO-7816-4

Određuje cijelu logičku strukturu pametne kartice te naredbe i protokole za komuniciranje, a specificira:

- sadržaj poruka, naredbi i odgovora, odaslanih od CAD uređaja prema kartici i obrnuto
- strukturu i sadržaj bajtova koje šalje pametna kartica kao odgovor na reset
- strukturu datoteka i podataka
- metode pristupa datotekama i podacima na kartici
- metode za sigurno komuniciranje
- metode pristupa algoritmima koji su procesuirani na kartici. Ova norma ne opisuje te algoritme

2.3.1.5. ISO-7816-5

Opisuje brojevni sustav za prikaz identifikatora aplikacija AID-a (eng. *Application Identifiers*). Svaki AID ima dva dijela. Prvi dio je identifikator registriranog davatelja aplikacije RID (eng. *Registered Application Provider Identifier*) koji se sastoji od 5 bajta i oni su jedinstveni (eng. *unique*). Drugi dio AID je varijabilne duljine do 11 bajtova koji omogućuje RID-u identifikaciju specifične aplikacije.

2.3.1.6. ISO-7816-6

Norma opisuje međuindustrijske podatkovne elemente, fizički transport uređaja i prijenos podataka, odgovor na reset ATR (eng. *Answer To Reset*). Ova specifikacija dozvoljava dva protokola T=0 i T=1. Kartica može podržavati bilo koji, ali ne oba.

2.3.2. EMV norma

EMV norma jedna je od važnijih normi za pametne kartice kojeg su formirale vodeće financijske tvrtke u kartičnom poslovanju Europay, Mastercard i Visa. Norma pokriva elektromehaničke karakteristike, protokole, podatke te instrukcije koje spaja sa bankovnim transakcijama. Cilj EMV specifikacije je da svi sustavi za plaćanje dijele iste POS-terminale (eng. *Point of Sales*), kao što to čine magnetne kartice i aplikacije za njih. Zbog toga što će se uskoro sve bankovne magnetne kartice mijenjati pametnim karticama, ova norma osigurava da nove bankovne pametne kartice budu kompatibilne s bankovnim transakcijskim sustavima. Zahvaljujući ovoj normi, sve bankovno orijentirane pametne kartice bit će kompatibilne jedna s drugom isto kao i prijašnje (dosadašnje) magnetne kartice. Fleksibilnošću EMV norme, bankama je dopušteno da dodaju svoje opcije i specijalne potrebe u platni sustav pametnih kartica.

2.3.3. GSM norma

GSM (eng. *Global Standard for Mobile Communications*) norma je jedna od najvažnijih normi za pametne kartice u digitalnim mobilnim komunikacijama. GSM specifikacija započela je 1982. godine pod CEPT-om (Conference Euopeanne des Postes et Telecommunications), kasnije je nastavljena pod ETSI-em (European Telecommunications Standards Institute). GSM specifikacija podijeljena je u dva dijela, prvi dio opisuje osnovne funkcionalne karakteristike dok drugi dio opisuje sučelje te logičku strukturu kartice. Originalno, ova specifikacija je bila prvenstveno napravljena za mobilnu telefonsku mrežu. Kada se pametna kartica počela koristiti u mobilnim telefonima kao modul identifikacije pretplatnika (eng. *Subscriber Identification Module – SIM*), dio GSM norme postala je norma pametnih kartica.

2.4. Primjena pametnih kartica

S velikim rastom internet tehnologije i elektroničke trgovine, pametne kartice su postale sve šire prihvaćene i korištene kao kartice sa sigurno spremljenim sadržajem. U ovom poglavlju opisat će se neke aplikacije gdje se koristi tehnologija pametnih kartica. Ove aplikacije možemo podijeliti u pet glavnih kategorija:

1. Elektroničko plaćanje
2. Računalna sigurnost

3. Telekomunikacije
4. Zdrastvena kartica
5. Transport

2.4.1. Elektroničko plaćanje

Kod elektroničkog plaćanja pametna kartica služi za spremanje novčanih sredstava korisnika u obliku elektroničkog novca ili elektroničkih kovanica ili žetona (eng. *tokens*). Elektronički novac može se koristiti za manje kupnje kod kojeg nebi bilo potrebe za autorizaciju preko PIN broja (eng. *Personal Identification Number*). Novčana sredstva na kartici pokriva korisnikova banka preko njegovog računa u banci ili se ta sredstva pokrivaju na neki drugi način. Kada korisnik plaća pametnom karticom elektronički novac se skida s pametne kartice i šalje na račun davaoca usluge. Korisnik može napuniti svoju karticu elektroničkim novcem u banci u bilo koje vrijeme.

Od 1994. godine došlo je do značajnijeg razvoja aplikacija za rad s elektroničkim novcem u Europi koje su se proširile i izvan Europe. U vezi pametnih kartica i elektroničkog novca bilo je razvijeno nekoliko globalnih projekata kao npr.: ProtonCard tvrtke Banksys, VisaCash od tvrtke Visa International te Mondex tvrtke Mastercard. Ovi projekti bili su širom prihvaćeni u trgovinama diljem svijeta.

2.4.2. Računalna sigurnost

Pametna kartica našla je veliku primjenu u računalnoj sigurnosti gdje se koristi za autentifikaciju korisnika u sustav, za spremanje digitalnih certifikata i kriptografskih ključeva (npr. javni i tajni ključ za asimetrične algoritme) te za kriptiranje i dekriptiranje poruka.

Jedan od primjera korištenja pametne kartice u računalnoj sigurnosti je korištenje pametne kartice u PKI sustavu gdje pametna kartica sprema javni i tajni ključ korisnika koji se tada koriste za kriptiranje i dekriptiranje poruka, kreiranje digitalnog potpisa i za spremanje digitalnih certifikata.

2.4.3. Telekomunikacije

Telekomunikacije su jedan od najvećih korisnika pametnih kartica. Od 1988. godine pametne kartice postale su jedne od najvažnijih komponenata u telefonskim stanicama (mobilni telefoni). Podaci mreže, informacije pretplatnika i svi kritični podaci mobilne mreže spremljeni su unutar kartice. S ovom karticom, pretplatnik može obaviti poziv s bilo kojeg mobilnog telefona. Također svaki poziv preko telefona može biti kriptiran što osigurava privatnost.

2.4.4. Zdrastvena kartica

Zbog visoke razine sigurnosti za spremljene podatke, pametne kartice nude novu perspektivu za zdrastvene aplikacije. Pametne kartice mogu se koristiti za spremanje različitih informacija o vlasniku kartice kao što su

osobne informacije vlasnika, polica osiguranja, hitne medicinske informacije, podaci o prijemu u bolnicu te tekući medicinski podaci o vlasniku.

Pametne kartice mogu se koristiti za spremanje informacija različitih razina koje su autorizirane pojedinim korisnicima. Liječnici mogu imati pristup medicinskim kartonu pacijenta na kartici. Podaci za hitne slučajeve kao npr. popis osoba kojima se treba javiti u slučaju nezgode te podaci o bolestima pacijenta koji mogu biti korišteni kako bi se spasio pacijentov život. U nekim zemljama potrebno je imati osiguranje za plaćanje bolnice, sa podacima o osiguranju spremljenim na kartici administrativna procedura je vrlo jednostavna.

2.4.5. Transport

Pametna kartica može se ponašati i kao elektronički novac za vozače koji trebaju platiti cestarinu prije nego mogu koristiti cestu, tunel ili most. Na pametnoj kartici se može povećati količina novca na naplatnoj stanici (npr. benzinska postaja), a kod svakog prolaza kroz neku kontrolnu točku iznos na kartici se smanjuje. Kod ove primjene najprikladnije bi bile bezkontaktne kartice jer njih nije potrebno umetati u CAD uređaj.

2.5. Komunikacijski model

Komunikacija između CAD uređaja (eng. *Card Acceptance Device*) i pametne kartice odvija se preko jedne linije i zbog toga se kartica i CAD uređaj moraju izmjenjivati u slanju poruka, dok jedna strana šalje poruku druga prima i obrnuto. Ovakav protokol zove se poludupleks (eng. *Half-Duplex*) protokol. Pametna kartica i CAD uređaj imaju gospodar-sluga (eng. *master–slave*) odnos gdje se kartica ponaša kao sluga, a CAD uređaj gospodar. Ovaj odnos označuje da kartica nikad ne šalje poruku bez vanjskog zahtjeva tj. komunikaciju uvijek pokreće CAD uređaj.

2.5.1. Protokoli za komuniciranje

Norma ISO-7816-3 specificira ukupno 16 protokola. Protokoli su označeni s 'T=' plus sekvencijalni broj od 0 do 15. U praksi nas zanimaju 2 protokola, a to su T=0 i T=1 protokol.

2.5.1.1. T=0 protokol

T=0 protokol je prvi normalizirani protokol za pametne kartice, a pravila kod kreiranja bila su minimalno korištenje memorije i maksimalna jednostavnost. Protokol je bajtovno orijentiran i svaki TPDU (eng. *Transmission Protocol Data Unit*) sastoji se dva dijela:

1. Zaglavlje
2. Podaci

Zaglavlje sadrži pet bajta kako je prikazano u tablici 2.2.

Ime	Opis
CLA	Specificira klasu instrukcije
INS	Specificira određenu instrukciju
P1	Specificira adresu
P2	Specificira adresu
P3	Specificira broj bajtova prenesenih prema ili od pametne kartice

Tablica 2.2. Elementi zaglavlja TPDU naredbe za T=0 protokol

Mehanizam provjere greške je jednostavan i koristi provjeru pariteta pojedinog bajta. Kod detekcije pogreške zahtijeva se ponovno slanje naredbe.

Kartica odgovara CAD uređaju sa zaglavljem koje sadrži četiri bajta kako je prikazano u tablici 2.3.

Ime	Opis
ACK	Označava primitak [CLA,INS] naredbe
NULL	Koristi se za kontrolu protoka I/O kanala
SW1	Status informacija za nadzor tekuće naredbe
SW2	Status informacija za nadzor tekuće naredbe(opcionalno)

Tablica 2.3. Elementi zaglavlja odgovora TPDU naredbe kod T=0 protokola

2.5.1.2. T=1 protokol

T=1 je blokovski orijentiran protokol. Ovo znači da se dobro definiran skup podataka ili blok prenosi kao jedna podatkovna jedinica između pametne kartice i CAD uređaja. U blok može biti ugrađena APDU (eng. *Application Protocol Data Unit*) naredba namijenjenja nekoj specifičnoj naredbi. Ova mogućnost omogućava vrlo dobro uslojavanje između sloja povezivanja i aplikacijskog sloja. Stavljanje informacije u blok zahtijeva da blok bude prenesen bez greške jer inače protokol može biti jednostavno izgubljen. Detekcija i ispravak grešaka kod T=1 protokola je zbog toga kompleksnije nego kod T=0 protokola.

Detekcija pogrešaka u T=1 protokolu se obavlja uzdužnom redundancijom znakova LRC (eng. *Longitudinal Redundancy Character*), koji je u suštini puno kompleksniji od provjere pariteta koja se obavlja u T=0 protokolu, ili se obavlja korištenjem cikličke redundancije znakova CRC (eng, C*yclic Redundancy Check*). Ove operacije provjere grešaka garantiraju detektiranje bilo koje jednobitne greške u prijenosnom bloku. Kada se

detektira greška koja je nastala tijekom prijenosa signalizira se odašiljatelju da ponovi slanje bloka kojem je detektirana greška.

U T=1 protokolu postoje tri osnovna različita tipa blokova koji imaju jednaku strukturu, kako je prikazano na slici 2.4., ali različitu svrhu:

1. Informacijski blok – Ovaj blok služui za prenošenje podataka između aplikacijskog programa u kartici i aplikacijskog programa na strani poslužitelja.
2. Blok potvrde i primanja – Ovaj blok služi za prenošenje negativne ili pozitivne potvrde o tome da li je blok prenesen bez greške preko informacijskog kanala.
3. Nadzorni blok – Ovaj blok služi za prijenos kontrolnih informacija između pametne kartice i CAD uređaja.

Uvodno polje Prologue field			Informacijsko polje Information field	Zaključno polje Epilogue field
Adresa čvora Node address NAD	Kontrolni bajt protokola Protocol control byte PCB	Duljina Length LEN	APDU	Detekcija pogreške Error detection LRC/CRC
1 bajt	1 bajt	1 bajt	0 do 254 bajta	1 do 2 bajta

Slika 2.4. Struktura T=1 prijenosnog bloka

Svaki T=1 blok kao što je vidljivo iz slike 2.4. ima tri polja:

- **Uvodno polje (eng. *prologue field*)** – Obavezno polje bloka koje ima duljinu od 3 bajta. Sadrži ova tri elementa:
  - **NAD** – Adresa čvora (eng. *node address*)
  - **PCB** – Kontrolni bajt protokola (eng. *Protocol control byte*)
  - **LEN** – duljina
- **Informacijsko polje (eng. *information field*)** – Opcionalno polje bloka koje može biti duljine do 254 bajta.
- **Zaključno polje (eng. *epilogue field*)** – Obavezno polje bloka koje je duljine od jednog do dva bajta.

NAD element se koristi za identifikaciju adrese izvora i odredišta bloka. Ova mogućnost adresiranja je od velike koristi kad se T=1 protokol koristi za podaržavanjej višestrukih logičkih veza između pametne kartice i višestrukih aplikacijskih spojnih točaka na strani CAD uređaja. NAD ima dva podelementa:

- SAD (eng. *Source Address*) – Adresa izvora označava se sa tri najmanje značajna bita NAD bajta.

- DAD (eng. *Destination Address*) – Adresa odredišta označava se od petog do sedmog bita NAD bajta

U situacijama gdje se ne koriste višestruki logički kanali NAD element postavljen je na nulu. Druga dva bita, koji se ne koriste za SAD i DAD podelemente, služe za prijenos informacija namijenjenih kontroliranju V_{PP} napajanja(napajanje za programiranje EEPROM memorije).

PCB element koristi se za identificiranje tipa bloka(informacijskog, bloka potvrde i primanja te nadzornog bloka). Dva najznačajnija bita PCB bajta koriste se za označavanje različitih tipova:

- Najznačajniji bit postavljen u 0 označava da je blok informacijski
- Dva najznačajnija bita postavljena u 1 označavaju da je blok nadzorni
- Najznačajniji biti postavljen u 1, a sljedeći postavljen u 0 označava da je riječ o bloku potvrede i primanja

2.6. APDU (eng. *Application Protocol Data Units*)

APDU (eng. *Application Protocol Data Units*) je internacionalno normalizirana jedinica podataka za razmjenu podataka između pametne kartice i CAD uređaja. U prijenosnom sloju se APDU naziva TPDU koji je u aplikacijskom sloju podijeljen na dva APDU dijela koji su: APDU naredba i APDU odgovor. APDU možemo opisati kućicama gdje svaka kućica sadrži naredbu koju šalje CAD uređaj pametnoj kartici ili pametna kartica šalje odgovor CAD uređaju.

2.6.1. APDU naredba

Svaka APDU naredba ima dva elementa zaglavlje i tijelo. Veličina zaglavlja je fiksna i iznosi 4 bajta, dok veličina tijela varira ovisno o količini podatak koji se šalju. Slika 2.5. prikazuje strukturu APDU naredbe.

APDU naredba						
Zaglavlje (obavezno)				Tijelo (opcionalno)		
CLA	INS	P1	P2	Lc	Podaci	Le

Slika 2.5. Struktura APDU naredbe

- **CLA** (1 bajt) – Ovo obavezno polje identificira aplikacijsko-specifičnu klasu instrukcije. Odgovarajuće CLA vrijednosti definirane su ISO-7816-4 normom. Tablica 2.4. prikazuje vrijednosti CLA polja i njihovu klasu instrukcija kojoj pripadaju.

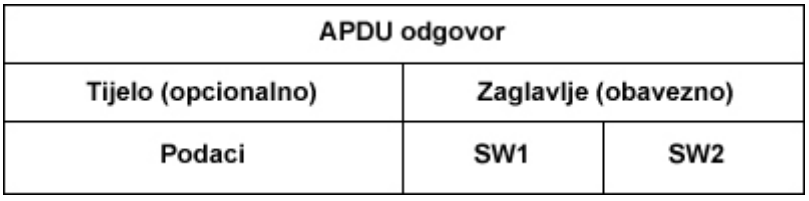
CLA vrijednost	Klasa instrukcije
0x0n, 0x1n	ISO-7816-4 instrukcije kartice, npr. za pristup datotekama i sigurne operacije
0x20 do 0x7F	Rezervirano
0x8n do 0x9n	ISO-7816-4 format koji se koristi za osobne aplikacijski-specifične instrukcije
0xAn	Aplikacijsko-specifične instrukcije
0xB0 do 0xCF	ISO-7816-4 format za korištenje aplikacijsko specifičnih instrukcija
0xD0 do 0xFE	Aplikacijsko-specifične instrukcije
FF	Rezervirano za izbor vrste protokola

Tablica 2.4. Vrijednosti i klasa CLA polja

- **INS** (1 bajt) – Ovo obavezno polje identificira specifičnu instrukciju sa klasom instrukcije definiranom u CLA polju. ISO-7816-4 norma specificira osnovne instrukcije za pristup podacima na pametnoj kartici. Korisnik može definirati svoje aplikacijsko-specifične instrukcije jedino kad koristi odgovarajuću vrijednost CLA polja.
- **P1** (1 bajt) – Ovo polje definira prvi parametar instrukcije. Može se koristiti za kvalifikaciju INS polja ili za ulazni podatak.
- **P2** (1 bajt) – Ovo polje definira drugi parametar instrukcije. Može se koristiti za kvalifikaciju INS polja ili za ulazni podatak.
- **Lc** (1 bajt) – Ovo opcionalno polje sadrži broj bajtova koji se nalaze u podatkovnom polju.
- **Podaci** (varijabilna veličina, Lc bajtova) – Opcionalno polje koje sadrži podatke naredbe.
- **Le** (1 bajt) – Opcionalno polje koje specificira maksimalni broj bajtova u podatkovnom polju očekivanog odgovora.

2.6.2. APDU odgovor

Struktura APDU odgovora koji vraća kartica je mnogo jednostavnija i prikazana je na slici 2.6.

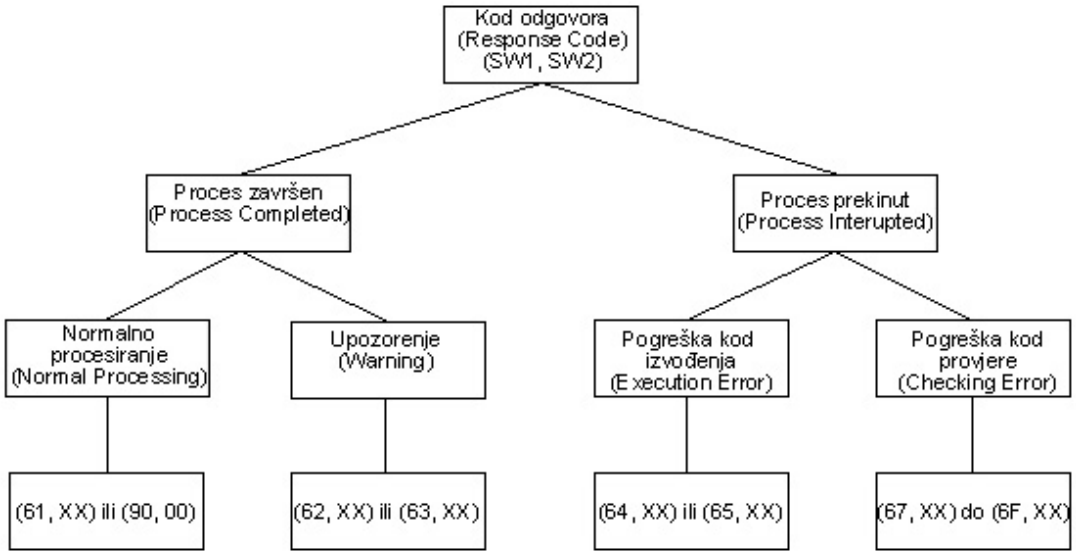


Slika 2.6. Struktura APDU odgovora

Također kao i APDU naredba i APDU odgovor ima opcionalna i obavezna polja.

- **Podaci** (varijabilna veličina, određene Le poljem u APDU naredbi) – Opcionalno polje koje sadrži podatke koje vraća aplikacija na kartici.
- **SW1** (1 bajt) – Obavezno polje sadrži statusnu riječ 1.
- **SW2** (1 bajt) – Obavezno polje sadrži statusnu riječ 2.

Vrijednosti statusne riječi definirane su ISO-7816-4 normom. Moguće vrijednosti statusne riječi SW1 i SW2 mogu se prikazati blok dijagramom prema načinu izvođenja procesa kako je prikazano na slici 2.7.



Slika 2.7. Blok dijagram statusnih riječi

3. Java Card tehnologija

Java Card tehnologija predstavlja najmanju Java platformu koja je namijenjena memorijski ograničenim uređajima poput pametnih kartica, te omogućuje da se programi pisani u Java programskom jeziku mogu izvršavati na takvim uređajima. Ova tehnologija je ustvari podskup Java tehnologije koji omogućuje kreiranje aplikacija za pametne kartice tzv. applete koji se izvršavaju na kartici.

Tipični Java Card uređaj ima mikroprocesor koji može biti 8-bitni, 16-bitni ili 32-bitni te sadrži 1KB RAM memorije i više te najmanje 16KB trajne memorije (EEPROM). Većina kartica također ima i posebne koprocesore te RAM memoriju za kriptografske algoritme.

Java Card tehnologiju možemo podijeliti na tri dijela:

1. Java Card stogovni stroj (eng. *Java Card Virtual Machine, JCVM*
2. Java Card izvršna okolina (eng. *Java Card Runtime Environment, JCRE*)
3. Java Card programsko sučelje (eng. *Application Programming Interface, API)*

3.1. Java Card stogovni stroj

Java Card stogovni stroj (eng. *Java Card Virtual Machine, JCVM*) specifikacija definira podskup Java programskog jezika i Java kompatibilnog virtualnog stogovnog stroja (eng. *Virtual Machine, VM*) za pametne kartice uključujući prikaz binarnih podataka i formate datoteka te JCVM set instrukcija.

VM za Java Card platformu implementiran je u dva dijela. Prvi dio je razvojni alat, tipično prikazan kao Java Card Converter tool, koji učitava, verificira te dalje priprema Java razrede u Java paketu za izvršavanje na pametnoj kartici. Izlaz iz konvertera je konvertirana applet datoteka ili CAP (eng. *Converted Applet*) datoteka koja se učitava na karticu. CAP datoteka sadrži sve razrede iz Java paketa u binarnom obliku za izvršavanje na kartici. Drugi dio VM-a je implementiran na kartici i on interpretira bajtkod, rukovodi razredima i objektima itd.

JCVM podržava samo ograničen podskup Java programskog jezika, ali sadrži mnoge poznate mogućnosti uključujući objekte, nasljeđivanje, pakete, dinamičko kreiranje objekata, virtualne metode, sučelja te iznimke. JCVM specifikacija izbacuje potporu za mnoge elemente Java jezika koji bi koristili previše memorije na ionako ograničenoj memoriji pametne kartice. Tablica 3.1. prikazuje pregled ograničenja Java Card programskog jezika dok tablica 3.2. prikazuje pregled ograničenja Java Card stogovnog stroja (JCVM).

Mogućnosti jezika	Dinamičko učitavanje razreda, sigurnosni manager, dretve, kloniranje objekata te neki aspekti kontrole pristupa paketima nisu podržani.
Ključne riječi (eng. *keywords*)	`native, synchronized, transient, volatile, strictfp` nisu podržane
Tipovi podataka	Nisu podržani `char, double, float` i `long` tipovi te nisu podržana višedimenzionalna polja. Podrška za `int` tip je opcionalna.
Razredi i sučelja (eng. *classes and interfaces*)	Razredi i sučelja jezgre Java API-a (`java.io, java.lang, java.util`) nisu podržani s iznimkom razreda `Object` i `Throwable`, ali većina njihovih metoda nije dostupna.
Iznimke (eng. *Exceptions*)	Neke podklase iznimaka (Exception) i grešaka (Error) su izostavljene jer ne mogu proizlaziti iz Java Card platforme.

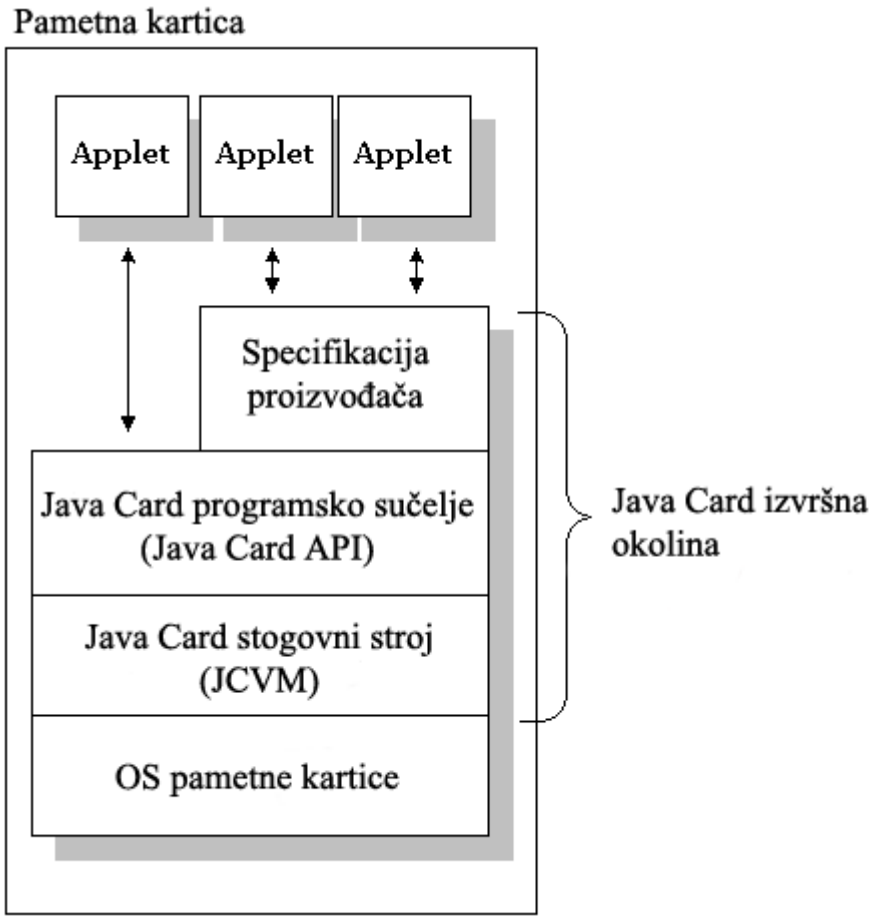
Tablica 3.1. Ograničenja Java Card programskog jezika

Paketi	Paket može sadržavati do 128 drugih paketa.
	Puno ime paketa je ograničeno na 255 bajta.
	Paket može imati do 255 razreda.
Razredi	Razred može direktno ili indirektno implementirati do 15 sučelja.
	Sučelje može naslijediti do 14 sučelja.
	Paket može imati do 26 statičkih metoda ako sadrži applet ili 255 ako ne sadrži.
	Razred može implementirati do 128 javnih ili zaštićenih metoda.

Tablica 3.2. Pregled ograničenja Java Card stogovnog stroja

3.2. Java Card izvršna okolina

Java Card izvršna okolina (eng. *Java Card Runtime Environment, JCRE*) specifikacija definira životni ciklus Java Card stogovnog stroja (JCVM), životni ciklus appleta, selektiranje appleta, transakcije te dijeljenje i postojanost objekta. JCRE osigurava sučelje neovisno o platformi tj. Neovisnost o operacijskom sustavu kartice. JCRE se sastoji od Java Card stogovnog stroja (JCVM), Java Card programskog sučelja (Java Card API) te bilo koje specifikacije isporučitelja ili proizvođača pametne kartice. Slika 3.1. prikazuje arhitekturu Java pametne kartice.



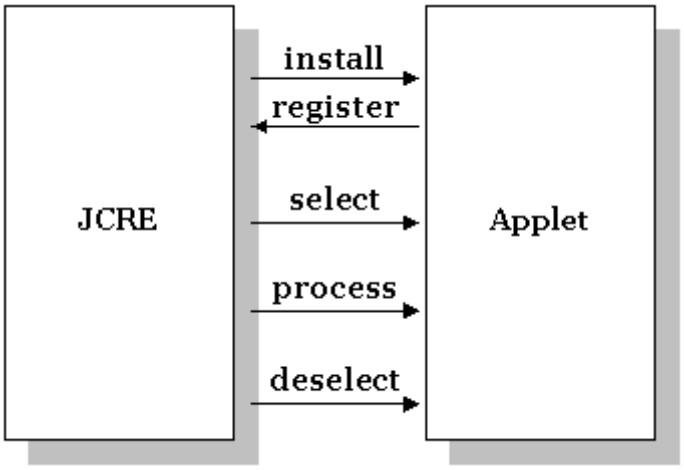
Slika 3.1. Arhitektura java pametne kartice

3.2.1. Životni ciklus Java Card stogovnog stroja

Životni ciklus Java Card stogovnog stroja (JCVM) podudara se s životnim ciklusom pametne kartice. Započinje nakon što je kartica proizvedena i testirana, prije nego je izdana korisniku, završava kada je odbačena ili uništena. JCVM se ne zaustavlja kad kartica nema napajanja zbog toga što je stanje JCVM-a spremljeno u trajnu memoriju pametne kartice. Nakon što je JCVM pokrenut sve interakcije s karticom su u principu kontrolirane jednim od appleta na pametnoj kartici. Kad se napajanje makne s kartice svi podaci koji su bili u RAM memoriji su uzgubljeni, međutim bilo koje stanje spremljeno u trajnu memoriju je sačuvano.

3.2.2. Životni ciklus Java Card appleta

Svaki aplet na kartici je jedinstveno identificiran aplikacijskim identfikacijskim brojem (eng. *Application ID, AID*). AID je sekvenca od 5 do 16 bajtova. Svaki aplet mora nasljeđivati `Applet` abstraktni razred koji definira metode koje koristi Java Card izvršna okolina (JCRE) za kontroliranje životnog ciklusa apleta kako je prikazano na slici 3.2.



Slika 3.2. Metode životnog ciklusa Java Card appleta

Životni ciklus Java Card appleta započinje kad je applet učitan na karticu i kad JCRE poziva statičku metodu appleta `Applet.install()` i kad se applet registrira sa JCRE-om pozivanjem `Applet.register()` metode. Jednom kad je applet instaliran i registriran on se nalazi u neselektiranom stanju, ali dostupan za selekciju i APDU procesiranje. U neselektiranom stanju applet je neaktivan. Applet dolazi u selektirano stanje za APDU procesiranje kada poslužiteljska aplikacija zahtijeva od JCRE-a da selektira određeni applet na kartici. Da JCRE obavijesti applet da je selektiran, od poslužiteljske aplikacije, JCRE poziva `select()` metodu. Applet tipično izvodi odgovarajuću inicijalizaciju kao pripremu za APDU procesiranje. Jednom kad je selekcija obavljena JCRE preusmjerava dolazeće APDU naredbe selektiranom appletu pozivajući metodu `process()`. Deselekcija appleta se izvodi kad poslužiteljska aplikacija zahtijeva od JCRE-a da selektira neki drugi aplet. JCRE obavještava aktivni applet da će biti deselektiran pozivajući `deselect()` metodu, koja tipično izvodi bilo koje logičko čišćenje memorije i vraća applet u neaktivno tj. neselektirano stanje.

3.3. Java Card programsko sučelje

Java Card programsko sučelje (API) definira mali podskup standardnog Java programskog sučelja. Ovo je najmanje programsko sučelje Java platforme.

U zamjenu za mali podskup sličnih razreda sa standardnom jezgrom Java programskog sučelja, Java Card API definira svoj skup razreda koji podržavaju Java Card aplikacije. Ovi razredi se nalaze u sljedećim razredima:

- `java.io` definira jedan razred iznimke, `IOException` razred, u svrhu da se kompletira hijerarhija poziva udaljenih procedura (eng. *Remote Method Invocation, RMI*) iznimaka.

- `java.lang` definira `Object` i `Throwable` razrede koji imaju velik nedostatak metoda naspram razreda u standardnom Java jeziku. Ovaj paket također definira nekolicinu razreda iznimaka, osnovni razred za iznimke `Exception` te različite iznimke kod izvođenja (eng. *Runtime Exception*).
- `java.rmi` definira `Remote` sučelje i razred `RemoteException`. Podrška za udaljeni poziv procedura (eng. *Remote Method Invocation, RMI*) je uključena da se pojednostavi migracija s uređajima koji koriste Java Card tehnologiju.
- `javacard.framework` definira sučelja, razrede i iznimke koji sačinjavaju jezgru Java Card sučelja. Definira važne koncepte poput osobnog identifikacijskog broja (eng. *Personal Identification Number, PIN*), `APDU` razred, Java Card `Applet` razred, te Java Card sustav (eng. Java Card System) `JCSystem` razred. Također definira razne konstante ISO7816 norme u razne specifične iznimke.
- `javacard.security` definira razrede i sučelja za Java Card sigurnosni okvir. Java Card specifikacija definira snažno sigurnosno sučelje koje sadrži različite tipove javnih i privatnih ključeva te kriptografske algoritme, algoritme za računanje sažetka poruka i algoritme za kreiranje digitalnog potpisa.
- `javacardx.crypto` je dodatni paket koji definira `KeyEncryption` sučelje i razred `Cipher`. `Cipher` razred je osnovni abstraktni razred kojeg moraju naslijediti svi kriptografski algoritmi.

4. OpenCard tehnologija

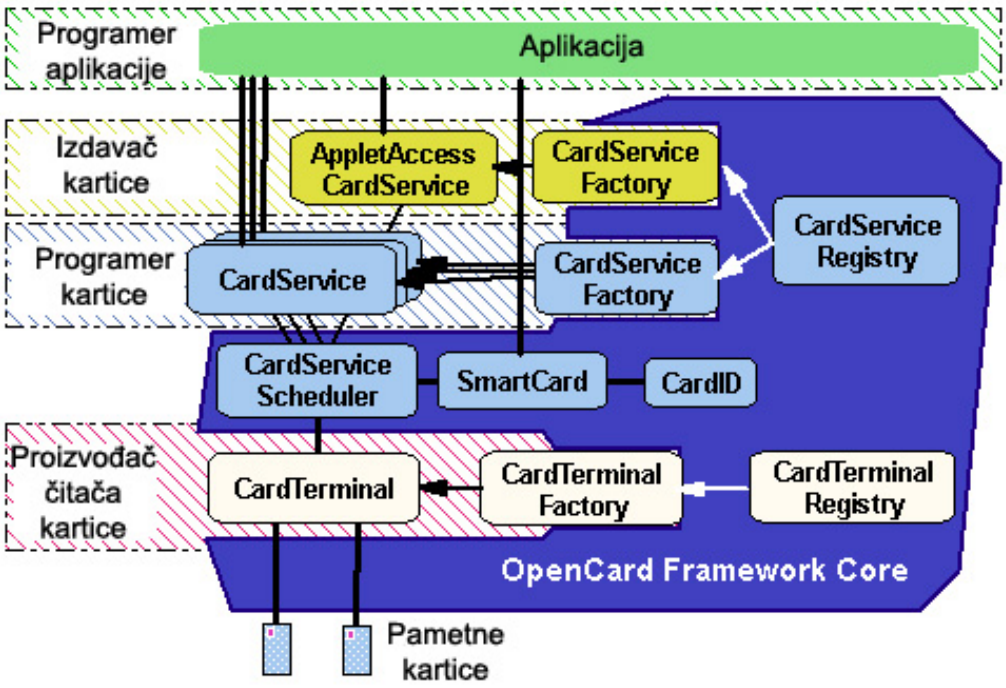
OpenCard tehnologija je otvorena arhitektura napravljena u programskom jeziku Java koja daje programsko sučelje koje omogućava komunikaciju s bilo kojom karticom i bilo kojim čitačem neovisno o proizvođaču. Velika prednost OpenCard tehnologije je u tome da se kod implementacije sustava za rad s pametnim karticama ne treba poznavati sklopovlje pametne kartice i čitača. Dio na višem sloju može se implementirati neovisno o nižim slojevima koji implementiraju komunikaciju s pametnom karticom.

4.1. Arhitektura OpenCard tehnologije

OpenCard implememtacija ima skup razreda i paketa koji su napravljeni da olakšaju i ubrzaju izradu sustava koji su temeljeni na pametnim karticama. Ova arhitektira omogućuje da:

- davatelj aplikacijskih i servisnih usluga
- davatelj kartičnog operacijskog sustava
- izdavač kartice
- proizvođač čitača kartice

rade zajedno.



Slika 4.1. Arhitektura OpenCard tehnologije

Davatelj aplikacijskih usluga izrađuje kartične i terminalske aplikacije koje vidi korisnik pametne kartice. Terminal koristi sučelja i kartične usluge OpenCard arhitekture kako bi komunicirao s kartičnom aplikacijom. Izdavač kartičnog operacijskog sustava izrađuje operativni sistem na koji upravlja pametnom karticom i izvodi programe na kartici. Izdavač kartice inicijalizira i izdaje pametnu karticu. Proizvođač čitača stvara uređaj koji komunicira s pametnom karticom. Slika 4.1. prikazuje arhitekturu OpenCard tehnologije.

Jezgra arhitekture OpenCard tehnologije sadrži dva glavna dijela:

1. Terminalski sloj (razred `CardTerminal`)
2. Servisni sloj (razred `CardService`)

4.1.1. Terminalski sloj

Terminalski sloj (`CardTerminal`) sadrži razrede i sučelja koji programeru aplikacije omogućuju pristup terminalima i njihovim priključcima (eng. *slots*). Na tržištu postoji velik broj čitačapametnih kartica s različitim funkcijama. Vrlo jednostavni čitači samo daju osnovne funkcije ulaza i izlaza (eng. *input-output*) preko jednog priključka, dok su drugi čitači sofisticiraniji i omogućuju više priključaka te mogućnost unosa PIN broja.

Razredi terminalskog sloja imaju dvostruku ulogu. Jedna od tih uloga je da omogućuje pristup fizičkom čitaču pametne kartice i umetnutoj kartici u taj čitač. Ove funkcije su enkapsulirane u `CardTerminal` razred preko priključaka (eng. *slots*) te preko `CardID` razreda.

`CardTerminal` razred je abstraktni razred kojeg nasljeđuje konkretni rezredi za određene terminale (čitači pametnih kartice). Pristup umetnutoj kartici dobiva se preko `SlotChannel` objekta. `CardTerminal` razred osigurava da je u svakom trenutku vremena instanciran maksimalno jedan `SlotChannel` objekt. To znači kad neki objekt dobije `SlotChannel` objekt nijedan drugi objekt ne može pristupiti pametnoj kartici tako dugo dok taj SlotChannel objekt nije oslobođen.

Terminalski sloj također ima mehanizam za dodavanje i odstranjivanje terminala. `CardTerminalFactory` razred i `CardTerminalRegistry` objekt implementiraju te funkcionalnosti. Svaki proivođač čitača pametnih kartica koji podržava OpenCard tehnologiju implementira `CardTerminalFactory` koji zna za određenu porodicu čitača, te implementira odgovarajući `CardTerminal` razred. `CardTerminalRegistry` objekt sadrži zapise o instaliranim čitačima pametnih kartica, te ima metode za prijavljivanje i odjavljivanje terminala.

CardTerminal razred također generira događaje i obavješćuje ostali dio OpenCard tehnologije kad je pametna kartica umetnuta ili izvađena iz terminala.

4.1.2. Servisni sloj

Servisni sloj omogućuje OpenCard tehnologiji rad s različitim operacijskim sustavima pametne kartice i raznim funkcijama koje one mogu ponuditi. `CardService` sučelje predstavlja programsko sučelje visoke razine prema aplikaciji i brine se o svim komunikacijskim detaljima APDU veze. Za stvaranje objekta `CardService` zadužen je razred `CardServiceFactory`. Pdatke o svim registriranim `CardServiceFactory` objektima sadrži razred `CardServiceRegistry`. `CardService` komunicira s pametnom karticom preko rezreda `CardServiceScheduler` koji služi za sinkronizaciju konkurentnih pristupa pametnoj kartici.

4.2. Rad s OpenCard tehnologijom

Prije korišenja fukcija iz OpenCard tehnologije mora se obaviti inicijalizacija. Kako bi se obavila inicijalizacija mora se prvo pozvati metoda `start()` iz razreda `SmartCard`, također bi trebalo i pozvati metodu `shutdown()` prije nego se završi aplikacija. Nakon inicijalizacije treba čekati da se pametna kartica umetne te da se tada može dobiti odgovarajući `CardService` objekt. Kod čekanja na pametnu karticu postoje dva načina. Jedan način je blokirajući i tada se rad programa zaustavlja tako dugo dok se kartica ne umetne, dok je drugi način neblokirajući te se program izvršava, a dok se kartica umetne generira se određeni događaj isto se događa kad se izvadi pametna kartica. Nakon što je pametna kartica umetnuta pristupa se instanciranju servisa (`CardService`) za rad s pametnom karticom.

Kako bi neki razred mogao koristiti neblokirajući način čekanja na pametnu karticu on mora implementirati `CTListener` sučelje koje ima sadrži dvije metode `cardInserted()` i `cardRemoved()`. Također treba prije korištenja te funkcionalnosti inicijalizirati generator događaja `EventGenerator`. Slika 4.2. prikazuje inicijalizaciju OpenCard tehnologije.

```
//inicijalizacija EventManagera te OpenCard tehnologije
public void start() {
      EventGenerator.getGenerator().addCTListener(this);
      try {
            SmartCard.start();
            EventGenerator.getGenerator().
                  createEventsForPresentCards(this);
      }
      catch (Exception e) {
            …
      }
}
```

Slika 4.2. Inicijalizacija OpenCard tehnologije

Nakon što inicijaliziramo OpenCard tehnologiju i nakon što je umetnuta pametan kartice pristupa se instanciranju `CardService` objekta. Instanca `CardService` objekta je `PassThruCardService` objekt koji omogućava komuniciranje s pametnom karticom na nivou APDU naredbi. Slika 4.3. prikazuje kako se instancira `PassThruCardService`.

```
…
CardRequest cardRequest = new CardRequest(CardRequest.ANYCARD,
      null, PassThruCardService.class);
SmartCard smartCard = SmartCard.getSmartCard(p_event, cardRequest);
PassThruCardService ptcs = smartCard.getCardService(
      PassThruCardService.class, true);
…
```

Slika 4.3. Instanciranje `PassThruCardService` razreda

5. Implementacija Java Card appleta

U ovom poglavlju opisuje se funkcionalnost i implementacija Java Card *appleta* za pametnu karticu napravljenog u praktičnom dijelu diplomskog zadatka. Kod izrade *appleta* velika pažnja se posvetila tome da se pametna kartica može koristiti u sustavu javnih ključeva (eng. *Public Key Infrastructure, PKI*), što bi značilo da može obavljati kriptiranje podataka, autentifikaciju korisnika, te spremanje digitalnih certifikata. Istodobno se kod izrade gledalo da taj *applet* nije jednonamjenski, samo za jednu arhitekturu i sustav već da se može iskoristiti i u drugim sustavima koji zahtijevaju neke druge funkcije od pametne kartice osim gore navedenih. U prvom dijelu ovog poglavlja prvo se opisuju funkcionalnosti koje ima *applet* za pametnu karticu, a nakon toga kako su te funkcionalnosti implementirane u Java Card tehnologiji.

5.1. Funkcionalnost appleta

Za mogućnost korištenja pametne kartice u PKI sustavu potrebno je da ona ima određene funkcionalnost. U Java Card tehnologiji funkcionalnosti implementira Java Card *applet* koji se izvršava na pametnoj kartici koja podržava Java Card tehnologiju. Implementiran *applet* ima sljedeće osnovne funkcionalnosti:

1. Autentifikacija i autorizacija
2. Kriptografija
3. Spremanje podataka

5.1.1. Autentifikacija i autorizacija

Autentifikacija i autorizacija je najbitnija funkcionalnost pametne kartice. Svaka pametna kartica ima svog krajnjeg korisnika koji koristi njezine funkcionalnosti te kako bi se osiguralo da baš vlasnik pametne kartice koristi zaštićene funkcionalnosti on se mora autentificirati, tj. mora dokazati da je baš on vlasnik te kartice. Autentifikacija se izvodi unosom PIN broja (eng. *Personal Identification Number*). Za dodatno osiguranje od neovlaštenog pristupa implementira se ograničen broj unosa pogrešnog PIN broja. Ovom metodom sprečava se da neovlaštena osoba s metodom promašaja i pokušaja otkrije PIN kartice te time neovlašteno koristi karticu. Pametna kartica implementira autorizaciju nad određenim funkcijama i podacima čime se sprečava da vlasnik kartice ne koristi neke funkcije koje su namijenjene administratoru kartice i obrnuto. Na kartici također postoje objekti (podaci) koje se može pročitati bez autentifikacije tj. ti objekti nemaju autorizacijskih ograničenja, već su to javni objekti koji mogu biti dostupni svima.

Implementirani *applet* ima dva PIN broja i dvije razine autorizacije. Za svaki od PIN brojeva postoji i PUK broj (eng. *Personal Unblocking Key*) koji omogućuje deblokiranje u slučaju da se PIN broj unese pogrešno, više puta nego je dozvoljeno. Ukoliko se i PUK broj unese pogrešno više puta nego je

dozvoljena kartica je blokirana i tada više ne postoji način da se kartica odblokira te je pristup podacima na kartici onemogućen.

Svaki od dva implemnentiran PIN broja ima svoju razinu autorizacije. Jedan PIN broj je korisnički i to je PIN kojim se autentificira korisnik kartice, dok je drugi PIN broj administratorski i njime se autentificira administrator kartice. Kad se korisnik autentificira on može kriptirati i dekriptirati podatke svojim javnim i tajnim ključem, mijenjati PIN broj, mijenjati PUK broj, čitati podatke s kartice za koje ima dozvole te može pisati podatke na karticu. Korisnik kartice ne može generirati ključeve te ne može inicijalizirati karticu, te funkcije može obavljati administrator kartice kad se autentificira svojim PIN brojem, međutim administrator ne može kriptirati podatke ključevima na kartici jer su to korisnički ključevi i samo ih on ima pravo koristiti.

Kako bi neovlaštenom korisniku bilo onemogućeno lagano otkrivanje PIN broja tada taj PIN broj treba biti što kompleksniji. Implementirani *applet* može imati PIN broj duljine 8 znakova (bajtova) koji mogu biti bilo koji alfanumerički znak. Pogrešan PIN broj može se maksimalmo unijeti tri puta. Za PUK broj implementirano je da duljina može biti 16 znakova (bajtova) koji također mogu biti bilo koji alfanumerički znak, dok je maksimalan broj pogrešnih unosa pet puta. Tablica 5.1. prikazuje atribute PIN i PUK brojeva.

	Duljina	**Maksimalan broj pogrešnog unosa**	**Prihvatljivi znakovi**
PIN	8 znakova (bajta)	3	Alfanumerički
PUK	16 znakova (bajta)	5	Alfanumerički

Tablica 5.1. Atributi PIN i PUK brojeva

5.1.2. Kriptografija

Druga funkcionalnost koju bi svaka pametna kartica trebala imati su kriptografske funkcije koje omogućuju pametnoj kartici kriptiranje i dekriptiranje podataka te generiranje ključeva.

Implementirani *applet* ima mogućnost kriptiranja i dekriptiranja podataka s asimetričnim RSA algoritmom duljine ključeva 1024 bita s nadopunom po PKCS#1 (eng. *Public Key Cryptography Standard*) normi. Pametna kartica ima mogućnost generiranje RSA para ključeva. Javni RSA ključ može se pročitati s kartice dok privatni ne može biti pročitan već se s njim mogu samo kriptirati (u slučaju digitalnog potpis) ili dekriptirati podaci. Za provjeru da pametna kartica sadrži odgovarajući privatni ključ za dani javni ključ implementiran je algoritam koji to provjerava.

5.1.2.1 Algoritam provjere privatnog ključa

Algoritam provjere privatnog ključa provjerava da li privatni ključ na pametnoj kartici odgovara javnom ključu kojeg ima terminal. Ovaj algoritam odvija se u pet koraka kako slijedi:

Ke - javni ključ koji je na terminalu

Kd - privatni ključ

1. terminal generira slučajni broj ***N***
2. terminal kriptira broj ***N*** javnim ključem ***Ke*** i šalje poruku $M_1 = E(N, Ke)$ kartici
3. kartica prima poruku M_1 te je dekriptira privatnim ključem ***Kd*** i dobiva broj $N = D(M_1, Ke)$
4. kartica uvećava ***N*** za jedan kriptira ga privatnim ključem ***Kd*** i šalje poruku $M_2 = E(N + 1, Kd)$ terminalu
5. terminal prihvaća poruku M_2 te je dekriptira javnim ključen ***Ke*** i dobiva iz nje broj $N + 1 = D(N + 1, Ke)$ te ako taj broj zadovoljava uvjet provjera privatnog ključa je prošla.

Za vrijeme izvršavanja ovog algoritma komunikacija između pametne kartice i terminala ne smije biti prekinuta jer u protivnom se smatra da provjera nije prošla.

5.1.3. Pohrana podataka

Funkcionalnost pohrane podataka na pametnu karticu daje mogućnost pohranjivanja određenih korisničkih podataka (npr. digitalni certifikati, podaci o korisniku) te čitanja pohranjenih podataka s pametne kartice.

Implementirani *applet* ima određenu količinu memorije u koju može spremati podatke. Osim ograničene memorije također je i ograničen broj objekata (datoteka) koji se mogu spremiti na pametnu karticu. U ovom slučaju kapacitet memorije za podatke je 16KB, dok je maksimalan broj objekata koji se mogu spremiti 32.

Svaki spremljen objekt na kartici je autoriziran s time da se odredi tko ima dozvolu za čitanje objekta, a tko ima dozvolu za brisanje objekta.

5.2. Implementacija appleta

Ovo poglavlje opisuje kako su navedene funkcionalnosti (poglavlje 5.1.) implementirane u Java Card tehnologiji.

Applet je implementiran u razredu `CardApplet` koji nasljeđuje razred `Applet` iz Java Card programskog sučelja. Metode koje `CardApplet` preuzima iz `Applet` razreda su `install()`, `select()`, `deselect()` i `process()`. Metoda `install()` poziva se kod instalacije appleta i ona se poziva svaki put kad se *applet* iznova instancira. Metoda `select()` se poziva kad selektiramo *applet* preko njegovog identifikacijsog broja (eng. *Application Identifier, AID*). Metoda `deselect()` se poziva kad se applet deselektira tj. kad se selektira applet s drugim AID brojem. Metoda `process()` je najvažnija metoda kod procesiranja naredbi koje pošalje terminal. Ta metoda služi da prima svaku pridošlu APDU naredbu koju je

poslao terminal te je obrađuje pozivajući privatne metode iz `CardApplet` razreda koje vraćaju APDU odgovor ovisno o rezultatu obrade.

5.2.1. Autentifikacija i autorizacija

Java Card tehnologija implementira razred `OwnerPIN` koji implementira funkcionalnost PIN broja. Ovaj razred omogućava promjenu i provjeru PIN broja, vraća broj preostalih pokušaja unosa PIN broja, te mogućnost odblokiravanja PIN broja čime se broj preostalih pokušaja vraća na maksimalni broj. Instanciranje PIN broja prikazuje slika 5.1. `CHV_TRY_LIMIT` predstavlja maksimalan broj pokušaja unosa PIN broja, a `CHV_SIZE` predstavlja duljina PIN broja u bajtovima.

```
private OwnerPIN m_userPin = new OwnerPIN(CHV_TRY_LIMIT, CHV_SIZE)
```

Slika 5.1. Instanciranje PIN broja

Funkcija deblokiranja PIN broja obavlja se tako da postoji i druga instanca `OwnerPIN` razreda koji ima funkciju PUK broja te provjerom tog broja i ovisno o rezultatu PIN broj se odblokirava i postavlja na novu vrijednost. Slika 5.2. prikazuje algoritam deblokiranja PIN broja.

```
if (m_userPuk.check(buffer, ISO7816.OFFSET_CDATA, pukLength)) {
      m_userPin.resetAndUnblock();
      m_userPin.update(buffer, bufferOffset, pinLength);
}
```

Slika 5.2. Deblokiranje PIN broja

Kod autorizacije provjerava se da li je PIN broj važeći te ako jest možemo izvršiti autoriziranu akciju. Slika 5.3. prikazuje provjeru autorizacije.

```
if (m_userPin.isValidated()) {
/* autorizirane akcije */
}
```

Slika 5.3. Provjera autorizacije

5.2.2. Kriptografski algoritmi

Kod Java Card pametnih kartica kriptografski algoritmi implementirani su na samoj pametnoj kartici u kriptografskim koprocesorima. Najčešći implementirani algoritmi su AES, DES kao simetrični te RSA kao asimetrični algoritam. Java Card tehnologija ima abstraktni razred `Cipher` koji instancira te algoritme. Za instanciranje određenog algoritma potrebno je pozvati

statičku metodu `getInstance` kako je prikazano na slici 5.4. `ALG_RSA_PKCS1` predstavlja tip algoritma dok drugi parametar koji je u ovom slučaju `false` označuje da li taj algoritam mogu koristiti drugi appleti na kartici tj. označuje se raspodijeljenost.

```
Cipher m_cipher = Cipher.getInstance(Cipher.ALG_RSA_PKCS1, false);
```

Slika 5.4. Instanciranje kriptografskog algoritma

Za mogućnost kriptiranja podataka potrebno je instancu `Cipher` razreda inicijalizirati kriptografskim ključem te modom koji određuje da li se podaci kriptiraju ili dekriptiraju. Slika 5.5. prikazuje kriptiranje podataka.

```
p_cipher.init(p_privateKey, Cipher.MODE_ENCRYPT);
p_cipher.doFinal(p_input, p_offset, p_length, p_output, 0);
```

Slika 5.5. Kriptiranje podataka

Osim kriptiranja Java Card tehnologija omogućuje generiranje kriptografskih ključeva te ima razrede koji implementiraju kriptografske ključeve. Implementirani applet kod kriptiranja koristi 1024 bitni RSA ključ. Prije samog korištenja ključ treba izgraditi čemu služi razred `KeyBuilder` i statička medota `buildKey` koja kao parametre koristi tip ključa, duljinu ključa te oznaku da li se ključ kriptira. U implementaciji se koristio tip za privatni (`TYPE_RSA_PRIVATE`) i javni (`TYPE_RSA_PUBLIC`) RSA ključ duljine 1024 bita (`LENGTH_RSA_1024`). Ova metoda samo izgrađuje ključ, ali ga ne inicijalizira niti ne generira. Za generiranje para ključeva služi razred `KeyPair`. Slika 5.6. prikazuje isječak koda koji služi za generiranje 1024 bitnog RSA para ključeva.

```
Keypair userKeyPair;
userKeyPair=new KeyPair(KeyPair.ALG_RSA,KeyBuilder.LENGTH_RSA_1024);
userKeyPair.genKeyPair();
```

Slika 5.6. Generiranje RSA para ključeva

5.2.3. Pohrana podataka

Kod spremanja podataka na pametnu karticu u Java Card tehnologiji nailazi se na problem da Java Card programsko sučelje nema nikakve razrede koji bi implementirali spremanje podataka u memoriju pametne kartice kao što to u standardnom Java jeziku čine razredi iz paketa `java.io`. Iz tog razloga moralo se pristupiti izradi razreda koji će to omogućiti.

5.2.3.1. Organizacija memorije za pohranu podataka

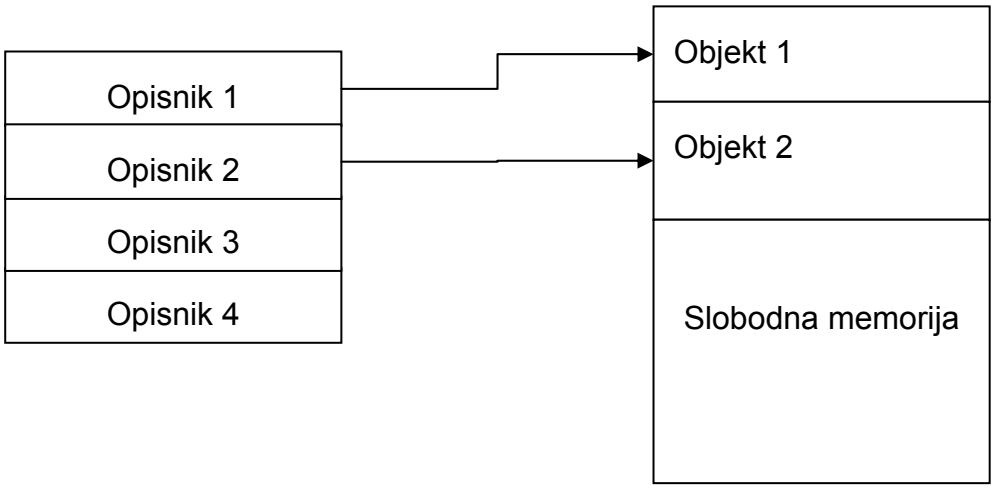
Kod izrade razreda koji će organizirati memoriju za pohranu podataka gledalo se na to da ti razredi omogućuju:

1. Pohranu podataka
2. Traženje podataka
3. Dohvaćanje podataka
4. Brisanje podataka

Kako bi se neki objekt spremljen u memoriji mogao pronaći tada on treba imati identifikator koji će nam omogućiti pronalaženje objekta u ovom slučaju to je 16 bitni broj i on mora biti jedinstven tj. ne mogu postojati dva objekta koja imaju isti identifikator. Za omogućavanje autorizacije podataka objekt treba imati podatak kojim se određuje tip autorizacije. Nadalje treba omogućiti defragmentiranje memorije nakon brisanja pojedinog objekta tako da je uvijek na raspolaganju maksimalna količina slobodne memorije. Kako bi se ove funkcionalnosti omogućile memorija je organizirana u obliku tablice gdje jedan objekt čini jedan elemenat u tablici. Uz tablicu objekata vezana je posebna tablica koja se sastoji od opisnika objekata u memoriji. Podatke koje sadrži opisnik objekta navedeni su u tablici 5.2. Slika 5.7. prikazuje strukturu organizacije objekata u memoriji.

Elementi opisnika objekta
Identifikator
Adresa početka
Veličina objekta u bajtovima
Autorizacija
Ime objekta

Tablica 5.2. Podaci opisnika objekta u memoriji



Slika 5.7. Struktura organizacije objekata u memoriji

Na slici 5.7. vidi se da su objekti u memoriji uvijek poredani tako da prvi opisnik pokazuje na prvi objekt u memoriji, drugi na drugi itd. Kad se podatak želi obrisati iz memorije obriše se samo opisnik te se napravi defragmentiranje tako da se svi objekti ispod objekta koji se obrisao pomaknu prema gore i pri tome se i opisnisi pomiču prema gore. Npr. ako imamo pet objekata u memoriji i obrišemo tećeg po redu, četvrti i peti objekt će se pomaknuti prema gore u tablici. Slijedi pseudokod ovog algoritma:

```
i = 0;
dok (i < brojOpisnika - 1) {
      ako je (opisnik[i] == null && opisnik[i + 1] == null) {
            return;
      }
      pomakni(opisnik[i + 1].objekt, opisnik[i].adresa)
      opisnik[i + 1].adresa = opisnik[i].adresa;
      opisnik[i] = opisnik[i + 1];
      opisnik[i + 1] = null;
}
```

Razred koji omogućuje spremanje, čitanje, brisanje i selektiranje objekta u memoriji je `JCHashTable`. Taj razred sadrži sljedeća polja:

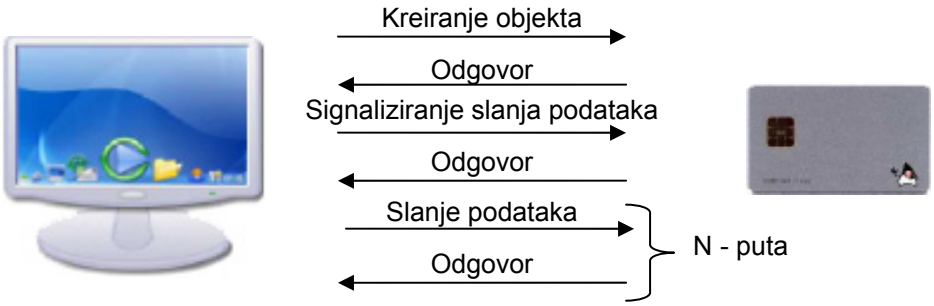
- memorija – bolje bajtova
- kapacitet – podatak o ukupnom kapacitetu memorije
- slobodna memorija – podatak o trenutno slobodnom prostoru u memoriji

- polje opisnika objekata – informacije o spremljenom objektu u memoriju

5.2.3.2. Pisanje, čitanje i brisanje objekta

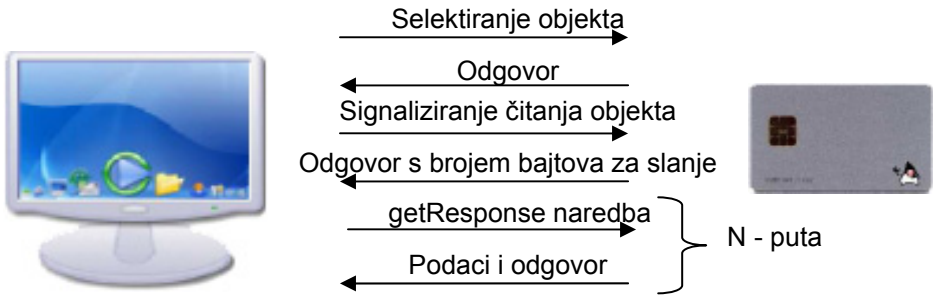
Kod čitanja i pisanja objekta na karticu javlja se problem da se jednom APDU naredbom ne može zapisati ni pročitati objekt koji je veći od 255 bajta jer jedna APDU naredba može maksimalno prenijeti 255 bajtova te su za te potrebe kreirane metode koje obavljaju višestruko pisanje i čitanje podataka. Za čitanje se koristi `getResponse()` metoda, a za pisanje `uploadData()` metoda. Međutim kako bi se ove metode mogle koristiti potrebno je prije poziva tih metoda napraviti odgovarajuće predakcije.

Kod pisanja objekta na pametnu karticu prvo treba kreirati opisnik objekta koji se sprema i rezervirati memoriju na pametnoj kartici u koju će se spremiti objekt, ako nema dovoljno memorije applet šalje terminalu odgovor kojim ga obavještava da nema dovoljno memorije dovoljno memorije za spremanje tog objekta. Kreiranje objekta obavlja metoda `createObject()` koja kreira novi opisnik u tablici opisnika i upisuje podatke u opisnik (tablica 5.2. prikazuje strukturu opisnika). Nakon što je objekt uspješno kreiran može se pristupiti učitavanju objekta pozivanjem metode `writeObject()` koja signalizira *appletu* da se pripremi za dobivanje podataka. *Applet* tada pripremi radnu tablicu u koju privremeno zapisuje dobivene podatke i šalje odgovor terminalu da može slati prvi blok s podacima. Terminal nakon toga šalje podatke APDU naredbom koju obrađuje metoda `uploadData()`. Za svaku dobivenu naredbu metoda zapisuje podatke u radnu tablicu te šalje odgovor terminalu da može slati sljedeći blok podataka. Nakon što pošalje sve podatke terminal signalizira *appletu* da je završio sa slanjem podataka tako što šalje naredbu bez podataka, nakon što je *applet* primio tu naredbu on sprema podatke iz radne tablice u tablicu objekata. Slika 5.8. prikazuje komunikaciju terminala i pametne kartice kod zapisivanja objekta na pametnu karticu.



Slika 5.8. Komunikacija prilikom zapisivanja objekta na pametnu karticu

Prilikom brisanja i čitanja objekta s kartice objekt treba prvo selektirati tako da se pošalje APDU naredba za selektiranje koja kao podatak šalje identifikator objekta. Ovaj APDU obrađuje metoda `selectObject()` koja tada pretraži tablicu opisnika objekata te ako objekt postoji vraća terminalu sadržaj opisnika, a ukoliko ne postoji šalje mu odgovarajući odgovor kojim mu to signalizira. Nakon što je objekt uspješno selektiran on se tada može obrisati ili pročitati. Brisanje se jednostavno obavlja tako da se zove metoda `deleteObject()` koja obriše opisnik selektiranog objekta u tablici opisnika i defragmentira memoriju. Prilikom čitanja šalje se naredba kojom se signalizira da će se podatak pročitati, tu naredbu obrađuje metoda `readObject()` koja tada kao odgovor vraća broj bajtova koje će poslati. Nakon toga terminal poziva metodu `getResponse()` koja šalje podatke, a u statusu odgovora šalje broj bajtova koje će slati u sljedećem pozivu. Nakon što nema više podataka za slanje šalje se odgovor terminalu da je slanje završilo. Slika 5.9. prikazuje komunikaciju kod čitanja objekta s kartice.



Slika 5.9. Komunikacija prilikom čitanja objekta s kartice

5.2.3.3. Autorizacija nad objektima

Kako bi se omogućilo da sve objekte ne mogu čitati i brisati svi korisnici potrebno je implementirati funkciju autorizacije nad objektima. Za omogućavanje toga svaki objekt treba imati u svom opisniku podatak kojim se određuje tko ga smije čitati, a tko brisati. Implementirani applet omogućuje osam razina autorizacije nad objektom spremljenim u memoriji appleta u smislu tko ga smije brisati i čitati (tablica 5.3.). Napomena: podatak na karticu može zapisati onaj korisnik koji se autentificira svojim PIN brojem.

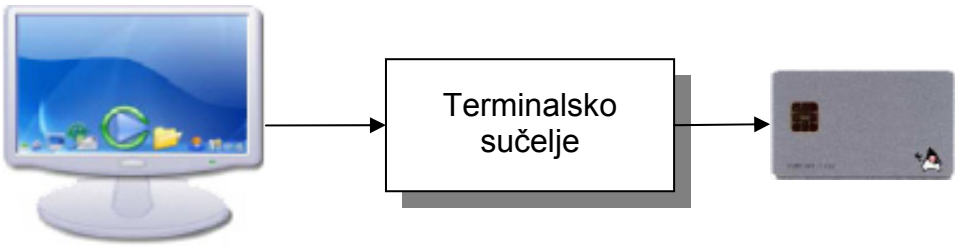
Vrijednost	Brisanje	Čitanje
0	Korisnik i administrator	Svi
1	Administrator	Svi
2	Korisnik kartice	Svi
3	Korisnik kartice	Korisnik kartice

4	Korisnik i administrator	Administrator
5	Korisnik i administrator	Korisnik kartice
6	Korisnik i administrator	Korisnik i administrator
7	Administrator	Administrator

Tablica 5.3. Autorizacija nad objektima

6. Implementacija terminalskog sučelja

Terminalsko sučelje služi za komuniciranje s implementiranim *appletom* na pametnoj kartici. Ovo programsko sučelje implementirano je tako da se može koristiti u bilo kojoj aplikaciji koja koristi implementirani *applet*. Dakle ovo sučelje je veza između aplikacije koja koristi pametnu karticu i pametne kartice kako prikazuje slika 6.1.



Slika 6.1. Terminalsko sučelje u komunikaciji između kartice i terminala

Terminalsko sučelje je implementirano u Java programskom jeziku korištenjem OpenCard tehnologije. Ovo sučelje podijeljeno je na veći broj mendžera koji rade određenu komunikaciju s karticom, opis menadžera je u poglavlju 6.1. Ostali bitni razredi i sučelja bit će opisani u poglavlju 6.2. Na kraju će biti prikazana i opisana aplikacija za administraciju pametne kartice.

6.1. Menadžeri terminalskog sučelja

Terminalsko sučelje nalazi se u `CardManager.jar` arhivi. Sučelja manadžera nalaze se u paketu `hr.rd.smartcard.manager`, a implementacije tih sučelja u `hr.rd.smartcard.manager.impl`. Menadžeri su podijeljeni ovisno o funkcionalnosti u zasebna sučelja. Tako ima sedam sučelja ovisno o namjeni:

1. SelectFileManager
2. InitializeCardManager
3. CardInformationManager
4. PinManager
5. PukManager
6. CryptoManager
7. CardObjectManager

6.1.1. SelectFileManager

Služi za selektiranje *appleta* na kartici. Sučelje ima jednu metodu `selectFile()` koja selektira applet na kartici. `SelectFileManagerImpl` razred implementira ovo sučelje.

6.1.2. InitializeCardManager

Služi za inicijalizaciju *appleta* na pametnoj kartici. Prije inicijalizacije *applet* je neupotrebljiv jer su tad sve funkcionalnosti appleta nedostupne. Ovaj manadžer ima metode:

- `initializeKeysOnCard()` - inicijalizira početne PIN i PUK brojeve na kartici.
- `setCardInformation()` - sprema informacije o pametnoj kartici i korisniku pametne kartice. Informacije koje sprema su identifikacijski broj kartice, datum isteka pametne kartice te DN (eng. *distinguished name*) korisnika kartice.
- `getCardInformation()` – čita informacije s pametne kartice.

  Razred `InitializeCardManagerImpl` implementira ovo sučelje.

6.1.3. CardInformationManager

Koristi se za dobavljanje informacija o ukupnom i slobodnom prostoru memorije. Ima dvije metode, prva je `getTotalMemorySize()`, a druga `getFreeMemorySize()`. `CardInformationManagerImpl` implementira ovo sučelje.

6.1.4. PinManager

Služi za promjenu, provjeru, postavljanje te deblokiranje PIN broja. Za svaku od ovih funkcija ima dvije metode, jedna je za rad s korisničkim PIN brojem dok se druga koristi za rad s administratorskim PIN brojem. Ovo sučelje implementira razred `PinManagerImpl`.

6.1.5. PukManager

Služi za promjenu, provjeru te postavljanje PUK broja. Također kao i `PinManager` ima za svaku od ovi funkcija dvije metode, jedna za rad s korisnikim PUK brojem dok je druga za rad s administratorskim PUK brojem. Razred koji implementira ovo sučelje je `PukManagerImpl`.

6.1.6. CryptoManager

Koristi se za korištenje kriptografskih funkcija pametne kartice. Metode koje ima ovo sučelje su:

- `cryptData()` – izvršavanje kriptiranja podataka na kartici. Kao parametre ova metoda dobiva bajtove podataka te mod kriptiranja. Mogu biti 4 moda kriptiranja:

    - Kriptiranje javnim ključem
    - Kriptiranje privatnim ključem
    - Dekriptiranje javnim ključem
    - Dekriptiranje privatnim ključem
- `generateKeyPair()` – zahtjev za generiranjem para ključeva na kartici.
- `getPublicKey()` – čitanje javnog ključa s pametne kartice.
- `verifyPublicKey()` – provjerava da li javni ključ odgovara privatnom ključu na pametnoj kartici (vidi poglavlje 5.1.2.1).

Ovo sučelje implementira razred `CryptoManagerImpl`.

6.1.7. CardObjectManager

Koristi se za čitanje, pisanje te brisanje objekta s pametne kartice. Metode koje ima ovo sučelje su:

- `createCardObject()` – kreiranje novog objekta na kartici, služi samo za kreiranje, ne upisuje podatke na karticu.
- `write()` – upisivanje objekta u memoriju pametne kartice.
- `read()` – čitanje selektiranog objekta s pametne kartice.
- `select()` – selektiranje određenog objekta na pametnoj kartici.
- `delete()` – brisanje selektiranog objekta s pametne kartice.

`CardObjectManagerImpl` implementira ovo sučelje.

6.2. Dodatni razredi i sučelja

Osim manadžera koji vrše komunikaciju s *appletom* na pametnoj kartici implementirani su i neki drugi razredi i sučelja koji olakšavaju kreiranje sustava za rad s pametnom karticom.

6.2.1. AbstractSmartCardApplet

Abstraktni koji razred nasljeđuje `JApplet` razred kojeg pokreće Java stogovni stroj u html pregledniku, te implemetnira `CTListener`. Funkcija ovog razreda je da inicijalizira objekte OpenCard tehnologije te da omogući neblokirajuć način čekanja na umetanje pametne kartice u terminal. Ovaj razred nasljeđuju *appleti* koji trebaju imati funkciju komuniciranja s pametnom karticom. U izradi ovog diplomskog zadatka to su *applet* za autentifikaciju korisnika u sustav digitalne knjižnice, *applet* za digitalno potpisivanje u digitalnoj knjižnici te *applet* za inicijalizaciju pametne kartice koji se koristi u implementaciji PKI sustava.

Osnovne metode koje ovaj applet implementira su:

- start() – inicijalizira objekte OpenCard tehnologije.

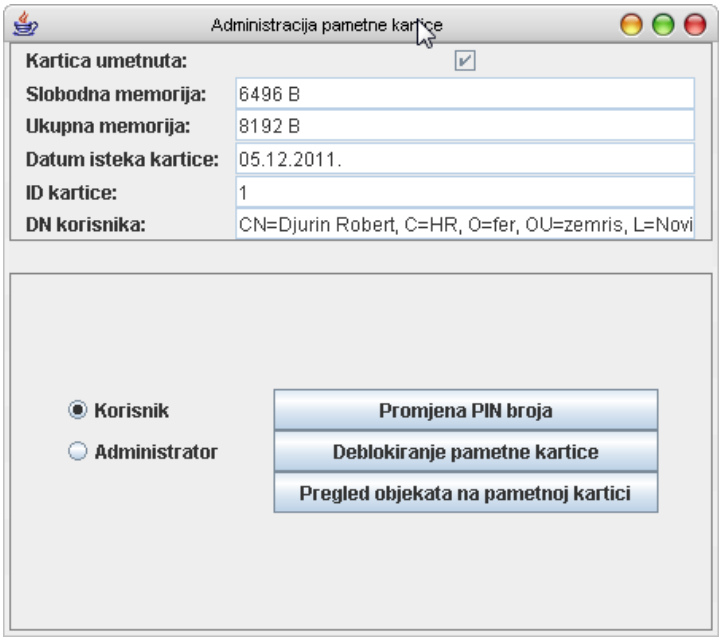
- stop() – uništava objekte OpenCard tehnologije.
- cardInserted() – poziva se kada je kartica umetnuta u čitač.
- cardRemoved() – poziva se kad je kartica izvađena iz čitača.

6.2.2. CardService

Ovo sučelje sadrži metode za rad s kriptografskim funkcijama pametne kartice, te za čitanje i pisanje digitalnih certifikata na pametnu karticu.

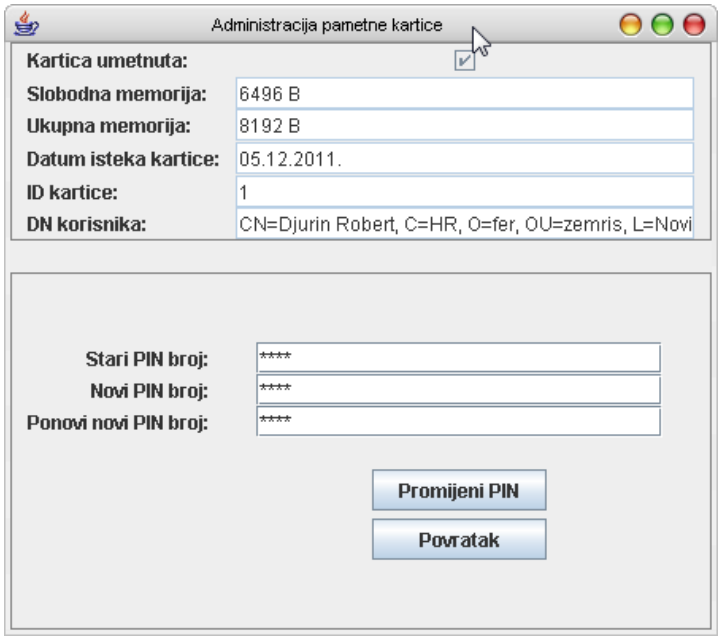
6.3. Aplikacija za administraciju pametne kartice

Ova aplikacija nudi mogućnosti promjene PIN broja, deblokiranje PIN broja te pregled objekata na pametnoj kartici. Također ispisuje informacije o pametnoj kartici te ukupnu i slobodnu memoriju pametne kartice. Ova aplikacija namijenjena je korisniku i administratoru pametne kartice. Slika 6.2. prkazuje uvodni ekran ove aplikacije. Ako kartica nije umetnuta tada su sve opcije onemogućene.

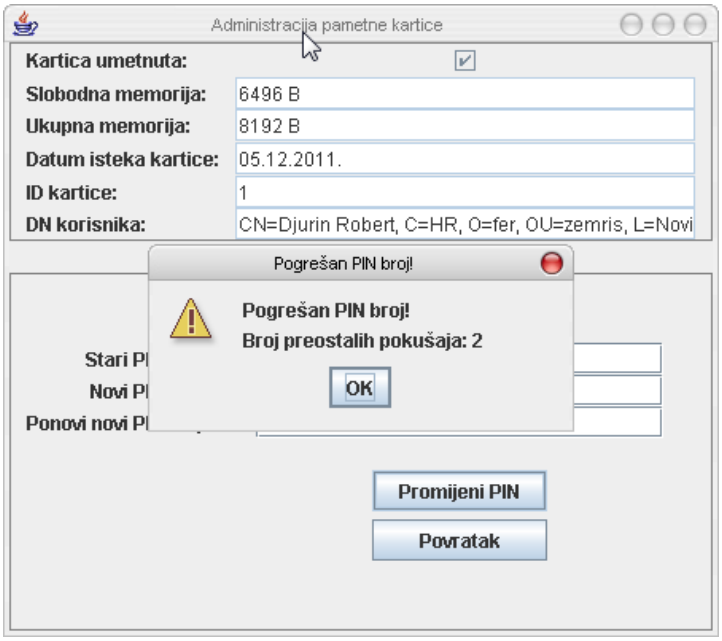


Slika 6.2. Uvodni ekran

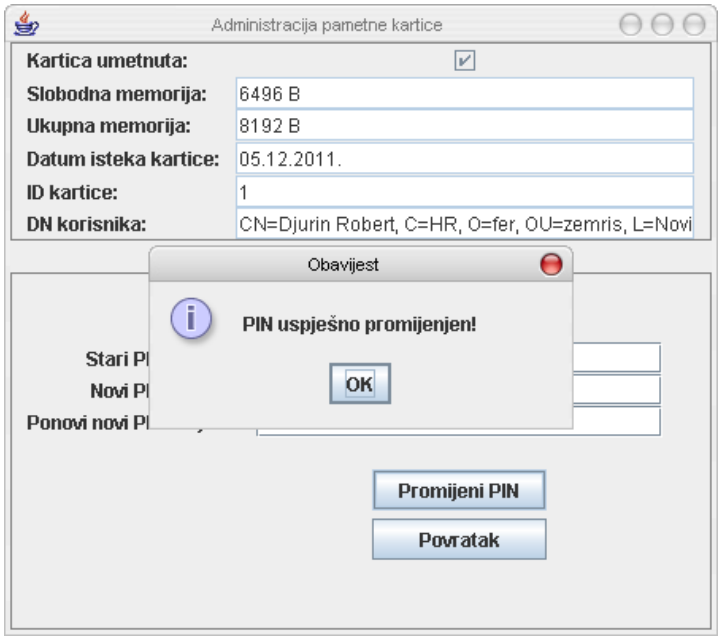
Ekran za promjenu PIN broja prikazan je na slici 6.3. Kod promjene PIN broja potrebno je unijeti stari PIN broj te dvaput novi PIN broj čime se sprečava greška unosa novog PIN broja. U slučaju da stari PIN broj ne odgovara javlja se poruka da je PIN broj krivo unesen te informacija o preostalom broju pokušaja kao što je vidljivo na slici 6.4. Ako je sve prošlo u redu javlja se ekran s informacijom da je PIN broj uspješno promijenjen (slika 6.5.).



Slika 6.3. Ekran za promjenu PIN broja

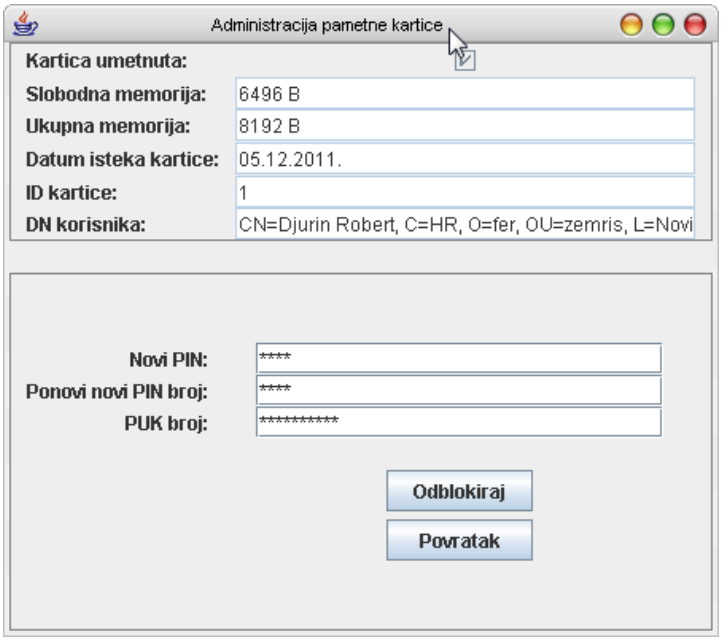


Slika 6.4. Pogrešno unesen stari PIN broj

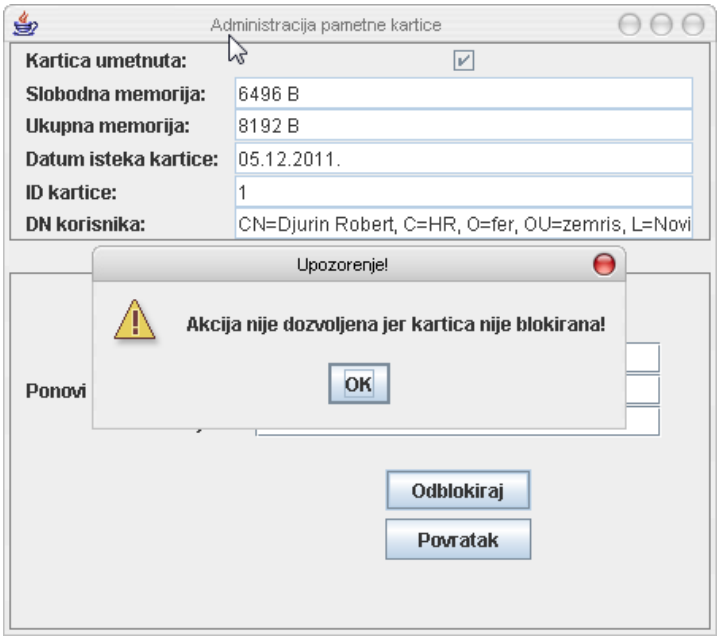


Slika 6.5. PIN broj uspješno promijenjen

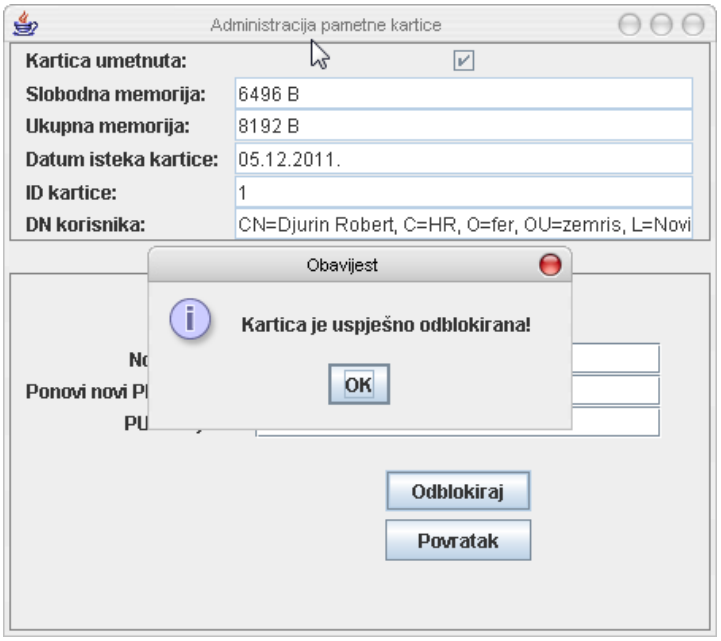
Kod promjene PUK broja unosi se dvaput novi PIN broj te PUK broj koji služi za odblokiravanje kartice. Ekran za odblokiravanje PIN broja prikazan je na slici 6.6. Ukoliko PIN broj nije blokiran ne dopušta se odblokiranje i javlja se informacija o tome (slika 6.7.). Ukoliko je PIN broj uspješno odblokiran javlja se informacija (slika 6.8.).



Slika 6.6. Ekran za odblokiravanje PIN broja

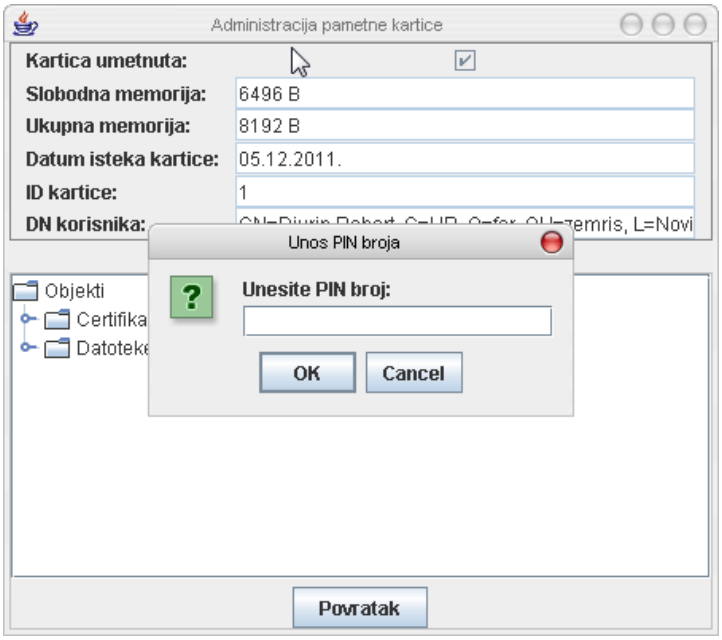


Slika 6.7. Informacija da kartica nije blokirana

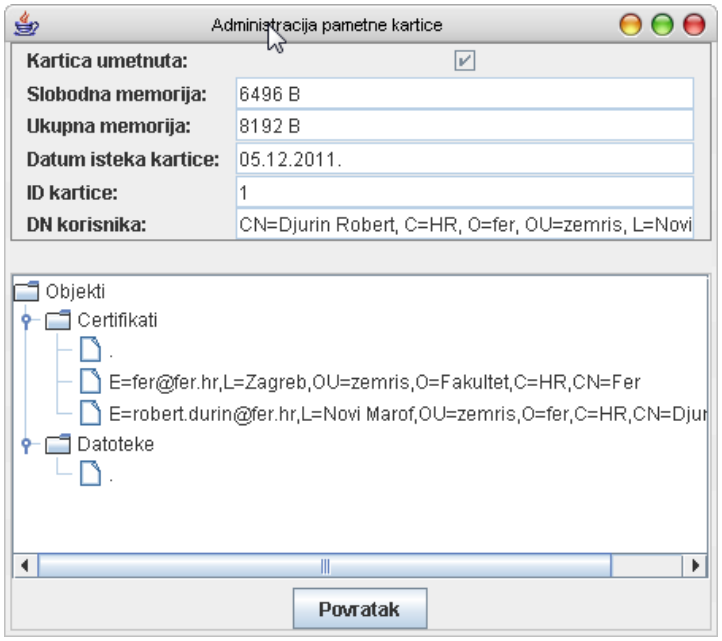


Slika 6.8. PIN broj uspješno odblokiran

Ekran za pregled podatak najprije zahtijeva unos PIN broja (Slika 6.9.) kako bi se podaci mogli pregledavati. Nakon uspješnog autentificiranja prikazuju se objekti s kartice svrstani u dvije podgrupe: certifikati i datoteke (slika 6.10.).



Slika 6.9. Unos PIN broja prije pregleda objekata na kartici



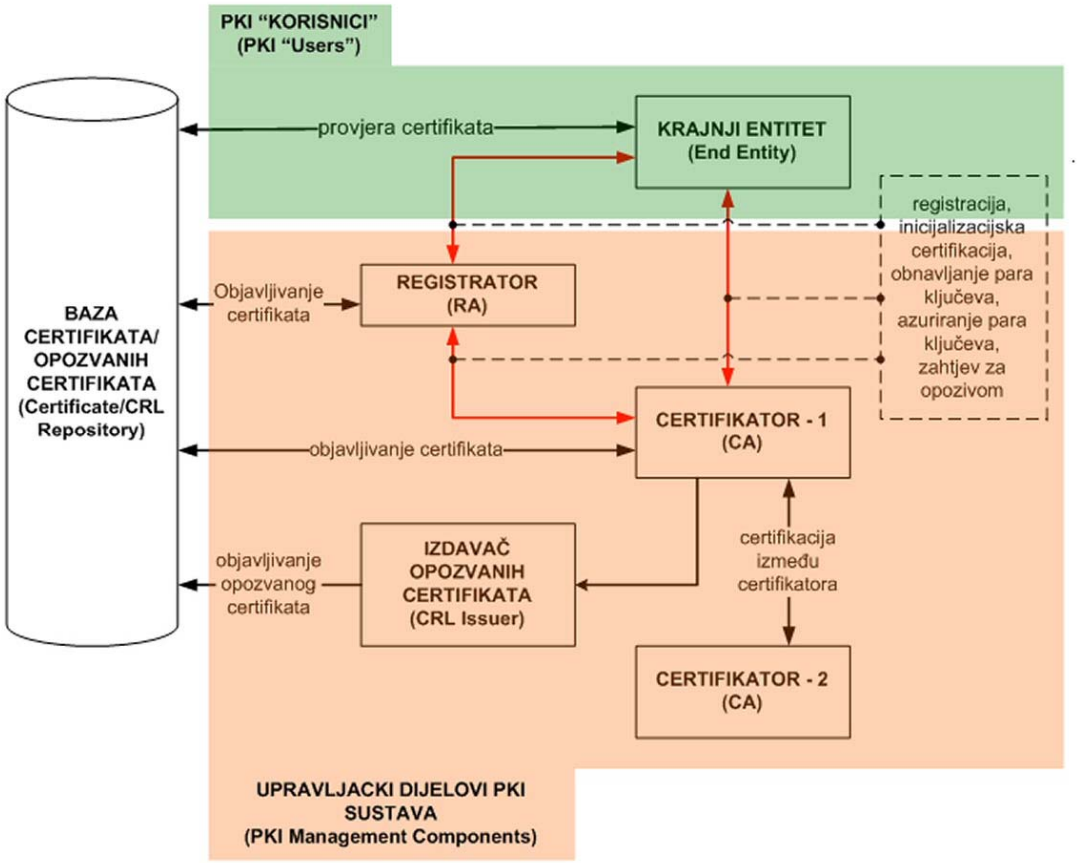
Slika 6.10. Pregled objekata na kartici

7. Sustav javnih ključeva (PKI)

Sustav javnih ključeva (eng. *Public Key Infrastructure, PKI*) je sustav zasnovan na asimetričnoj kriptografiji čija je svrha upravljanje ključevima i digitalnim certifikatima koji uspostavljaju povjerljivu okolinu te omogućuju povjerljivost podataka, integritet podataka, neporecivost te autentifikaciju. Funkcije PKI sustava su generiranje ključeva i certifikata krajnjim korisnicima, spremanje ključeva i certifikata, provjeravanje certifikata, registracija krajnjih korisnika te generiranje liste opozvanih certifikata (eng. *Certificate Revocation List, CRL*).

7.1. Arhitektura PKI sustava

Komponente PKI sustava (slika 7.1.) su certifikacijski centar (eng. *Certification Authority, CA*), registracijski centar (Registration Authority, RA), spremnik certifikata i opozvanih certifikata, te izdavač liste opozvanih certifikata (eng. *CRL Issuer*) te krajnji entiteti tj. korisnici PKI sustava.



Slika 7.1. Arhitektura PKI sustava

7.1.1. Certifikacijski centar (CA)

Funkcije certifikacijskog centra su izdavanje i potpisivanje digitalnih certifikata te opoziv digitalnih certifikata. Certifikacijski centar svojim potpisom jamči ispravnost podataka te valjanost certifikata kojeg je izdao. Certifikacijski

centar mora uspostaviti faktor povjeranja u sebe tako da svi ostali dijelovi imaju povjeranje u njega. Certifikacijski centar ima svoj privatni ključ kojim potpisuje certifikate koje je izdao, listu opozvanih certifikata te pripadajući javni ključ koji je spremljen u certifikatu tog certifikacijskog centra. Certifikacijski centar je temelj povjerenja u sustav te on mora biti najzaštićeniji dio sustava jer ako dođe do kompromitiranja privatnog ključa certifikacijskog centra tada dolazi do kompromitiranja cijelog sustava odnosno svih certifikata i liste opozvanih certifikata.

7.1.2. Registracijski centar (RA)

Registracijski centar ima funkciju registriranja krajnjih korisnika ili entiteta u PKI sustav, nadalje on može provjeravati da li korisnik ima privatni ključ koji odgovara javnom ključu koji ide u certifikat. Registracijski centar ne smije obavljati funkciju izdavanja i opozivanja certifikata. Registracijski centar može biti dio certifikacijskog centra ili može biti kao zasebni sustav.

7.1.3. Spremnik certifikata i opozvanih certifikata

Spremnik certifikata i opozvanih certifikata predstavlja sustav ili skup sustava koji pohranjuju certifikate i opozvane certifikate. Ovaj sustav mora biti dostupan svim unutarnjim sustavima, ali ujedno mora biti dostupan i vanjskim sustavima koji koriste certifikate te listu opozvanih certifikata.

7.1.4. Izdavač liste opozvanih certifikata

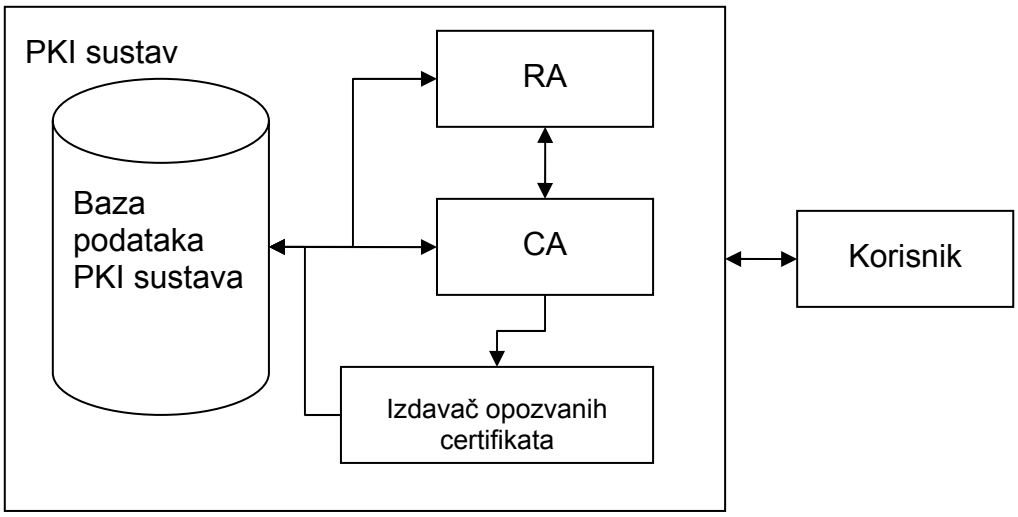
Izdavač liste opozvanih certifikata je dio PKI sustava koji izdaje listu opozvanih certifikata. Svaki certifikat ima svoje vrijeme trajanja koje se određuje vremenom od kada certifikat vrijedi te vremenom do kada certifikat vrijedi, međutim certifikat može biti opozvan i prije iz nekog razloga kao što je kompromitiranje privatnog ključa ili kompromitiranje privatnog ključa certifikacijskog centra.

7.1.5. Krajnji entiteti ili korisnici

Krajnji entiteti ili korisnici su vlasnici certifikata koje izdaje certifikacijski centar. Krajnji entitet ili korisnik ne mora biti fizička osoba već to može biti uređaj, poslužitelj ili aplikacija koji imaju certifikate da njima dokažu svoj identitet.

7.2. Implementacija PKI sustava

U okviru praktičnog rada implementiran je jednostavan PKI sustav. Arhitektura implmentiranog Pki sustava prikazana je slikom 7.2. Implementirani PKI sustav koristi RSA asimetrični algoritam s duljinom ključeva od 1024 bita kad se generira za korisnika sustava, odnosno 2048 bita kad se generira za certifikacijski centar. Kod potpisa certifikata i liste opozvanih certifikata (CRL) koristi se SHA1-RSA digitalni potpis. Certifikati i lista opozvanih certifikata se generiraju po X.509v3 normi.



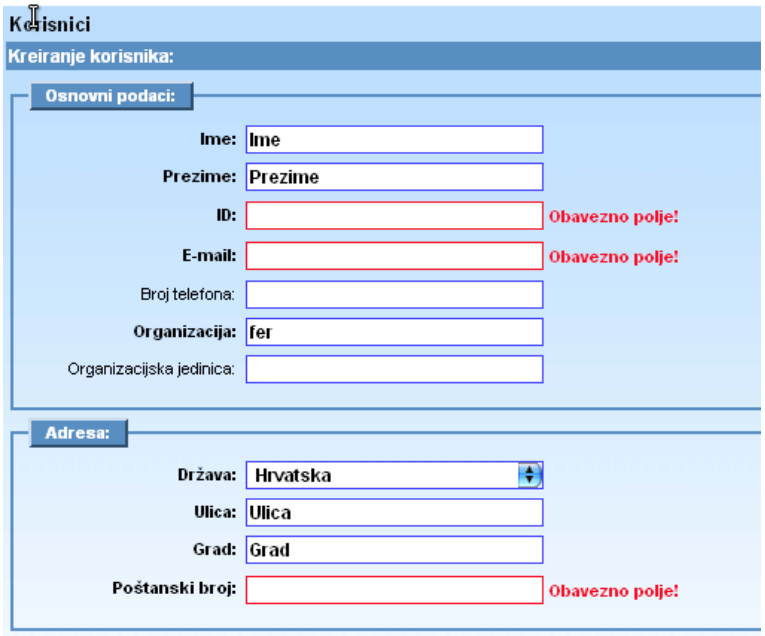
Slika 7.2. Prikaz implementiranog PKI sustava

Funkcionalnosti implemetiranog PKI sustava su:

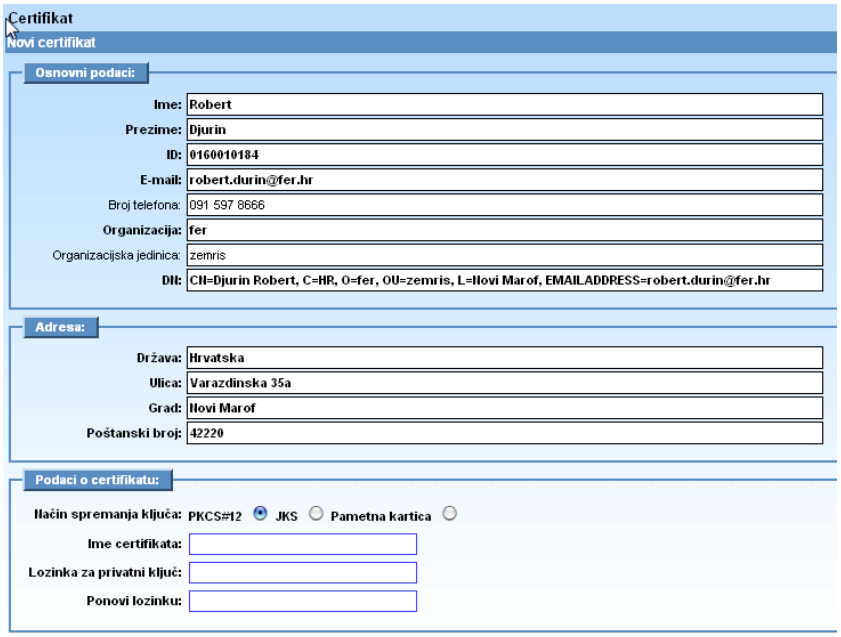
1. Registracija korisnika
2. Generiranje ključeva, certifikata i liste opozvanih certifikata
3. Spremanje i preuzimanje certifikata
4. Opoziv certifikata
5. Inicijalizacija pametne kartice
6. Administracija korisnika PKI sustava

Baza podataka PKI sustava služi za spremanje: certifikata, ključeva, podataka o korisnicima, certifikata certifikacijskog centra, ključeva certifikacijskog centra te podataka o korisnicima sustava. Budući su svi podaci spremljeni u jednu bazu podataka ta baza ne smije biti javna tj. za nju se mora osigurati nemogućnost neovlaštenog upada u bazu jer može doći do kompromitiranja privatnog ključa certifikacijskog centra. Stoga korisnik PKI sustava ne može čitati certifikat direktno iz baze već to može obaviti preko registracijskog centra ili u slučaju da treba dobaviti listu opozvanih certifikata to se obavlja preko certifikacijskog centra koji koristi usluge izdavača opozvanih certifikata. Tim pristupom kod implementacije sustava omogućava se neovlašten pristup bazi s podacima te mogućnost kompromitiranja sustava. U PKI sustav je implementiran poziv udaljenih procedura (eng. *Remote Method Invocation, RMI*) koji omogućuje drugim aplikacijama da koriste funkcionalnosti PKI sustava, a da se ne spajaju direktno na bazu podataka PKI sustava. Te procedure omogućuju dobavljanje liste opozvanih certifikata i dobavljanje certifikata.

Registracijski centar (RA) omogućuje registraciju korisnika i provjeru unesenih podataka (slika 7.3.). Nakon što su podaci ispravno uneseni korisnika se sprema u bazu podataka te je nakon toga njemu moguće generirati ključeve i certifikate (slika 7.4.).



Slika 7.3. Unos novog korisnika i provjera podataka



Slika 7.4. Generiranje novog certifikata

Generirane ključeve moguće je spremati na tri načina. Jedan način je po PKCS#12 normi spremanja ključeva, drugi način je u JKS (eng. Java Key Store) formatu. U slučaju ova dva načina potrebno je unijeti lozinku kojom se

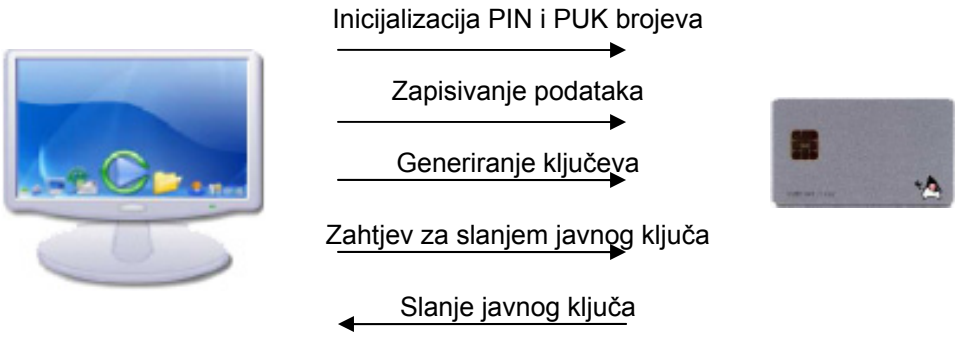
zaštićuje spremljeni privatni ključ od neovlaštenog pristupa, ova dva formata spremaju se u bazu podataka PKI sustava. Treći način spremanja ključeva je na pametnu karticu koje je detaljnije opisano u poglavlju 7.2.1.

Svaki certifikat koji je pohranjen u bazu podataka moguće je opozvati. Kod opoziva certifikata potrebno je odabrati razlog opoziva. Nakon što je certifikat opozvan on se nalazi na listi opozvanih certifikata. Listu opozvanih certifikata može se preuzeti kao datoteku. Također postoji i servis implementiran pomoću poziva udaljenih procedura koji omogućuje aplikacijama preuzimanje liste opozvanih certifikata. Sustav nudi mogućnost provjere valjanosti certifikata, to se obavlja slanjem certifikata PKI sustavu te nakon obrade sustav šalje rezultat o valjanosti certifikata.

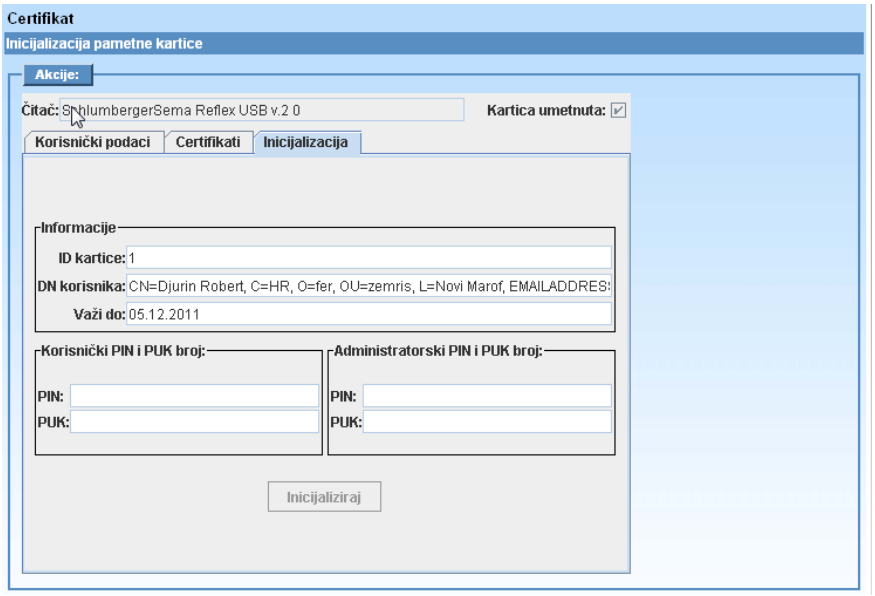
Kako bi se PKI sustav koristio za unos korisnika te generiranje certifikata u sustav se treba autentificirati. Time se bavi administracija korisnika sustava. Postoje dvije razine autorizacije, jedna je administratorska dok je druga korisnička. Administrator ima dodatnu funkciju administracije korisnika PKI sustava, dakle on dodaje ili briše korisnike koji mogu raditi s PKI aplikacijom.

7.2.1. Pametna kartica u PKI sustavu

Kada je pametna kartica spremište privatnog ključa tada ona generira ključeve te je privatni ključ pohranjen samo na pametnu karticu. U ovom slučaju PKI sustav vrši inicijalizaciju pametne kartice. Prilikom inicijalizacije postavljaju se korisnički i administratorski PIN i PUK brojevi, potom slijedi zapisivanje podataka o korisniku i kartici na pametnu karticu, ti podaci su DN (eng. *Distinguished Name*), identifikacijski broj pametne kartice te datum isteka pametne kartice. Potom terminal šalje zahtjev pametnoj kartici za generiranje RSA para ključeva te nakon toga preuzima javni ključ s pametne kartice koji će služiti za generiranje certifikata. Prije samog generiranje certifikata sustav prvo provjerava da li kartica ima odgovarajući privatni ključ, algoritam po kojem se to provjerava naveden je u poglavlju 5.1.2.1. Tok komuniciranja PKI sustava i pametne kartice kod inicijalizacije dan je na slici 7.5., a ekran koji prikazuje inicijalizaciju pametne kartice na slici 7.6.

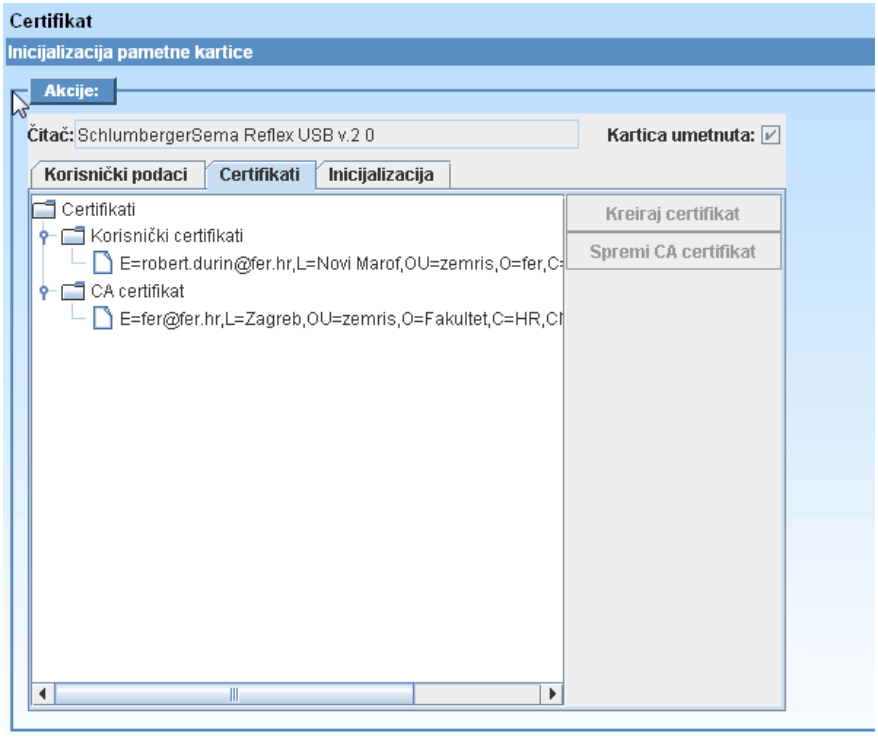


Slika 7.5. Komuniciranje kod inicijalizacije pametne kartice



Slika 7.6. Uspješna inicijalizacija pametne kartice

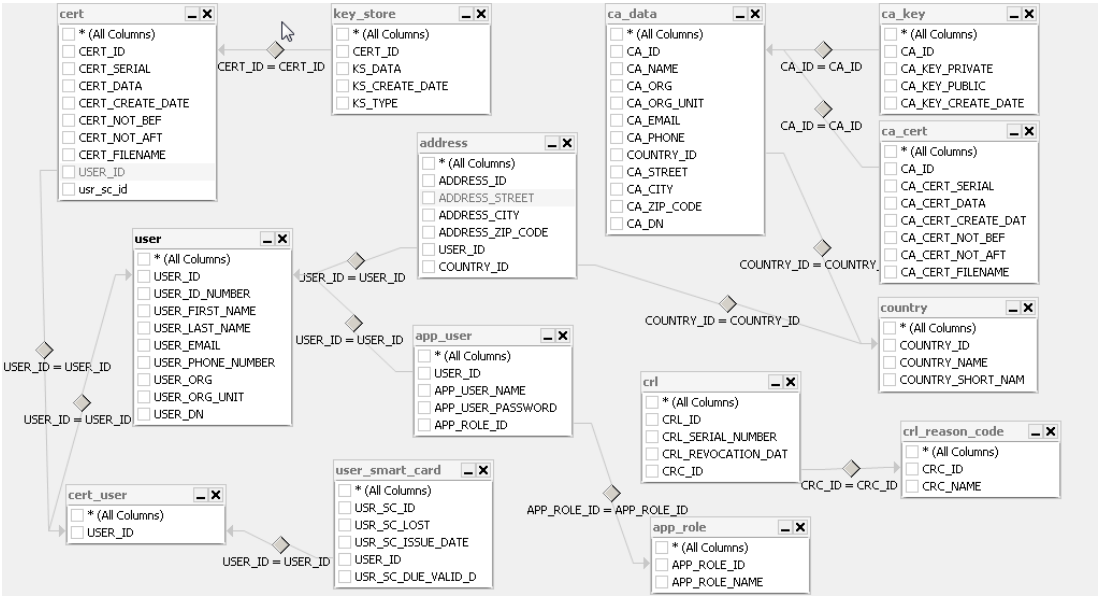
Nakon što je pametna kartica inicijalizirana na karticu se može pohranit korisnički certifikat te certifikat certifikacijskog centra. Ekran koji omogućuje tu akciju prikazan je na slici 7.7.



Slika 7.7. Spremljeni certifikati na pametnoj kartici

7.2.1. Baza podataka PKI sustava

Baza podataka realizirana je u MySql sustavu. Slika 7.8. prikazuje model baze podataka te veze među tablicama.



Slika 7.8. Model baze podataka za PKI sustav

Baza podataka podijeljena je na četiri grupe ovisno o podacima koje spremaju. Prvi dio sastoji se od tablica koje služe za spremanje podataka o certifikacijskom centru, drugi dio sastoji se od tablica za spremanje liste opozvanih certifikata, treći dio koji se sastoji od tablica za spremanje podataka o korisnicima i njihovih certifikata te četvrti dio služi za spremanje podataka o korisnicima sustava, također postoje i tablice koje koristi više grupa i one su tzv. šifrarnici. U bazi postoji 14 tablica. Slijedi opis tablica:

- CA_DATA – služi za spremanje osnovnih podataka o certifikacijskom centru. Ti podaci su ime, organizacija, organizacijska jedinica, email adresa, telefon, država, adresa te DN (eng. *distinguished name*).
- CA_KEY – služi za spremanje ključeva certifikacijskog centra, te uz to sprema i datum kreiranja ključeva. Ova tablica je povezana s CA_DATA tablicom preko stranog ključa. Relacija između ove dvije tablice je jedan naprema jedan.
- CA_CERT – sprema certifikat certifikacijskog centra te podatke o certifikatu koji su serijski broj certifikata, ime certifikata, datum kreiranja certifikata, datum od kada vrijedi certifikat te datum do kada vrijedi certifikat.
- USER –za spremanje podataka o korisnicima. U to spadaju korisnici certifikata te korisnici sustava. Sadrži podatke poput imena, prezimena, identifikacijskog broja, email adrese, telefonskog broja, organizacije, organizacijske jedinice, DN-a korisnika te adrese.

- ADDRESS – služi za spremanje adrese korisnika. Sprema ime grada, ulicu, poštanski broj. Povezana je s realacijom USER preko stranog ključa. Relacija između ovih tablica je jedan naprema jedan. Ova tablica je povezana stranim ključem s tablicom COUNTRY i relacija između ovih tablica je više naprema jedan.
- COUNTRY – služi za spremanje podataka o državama. Sprema podatke o imenu države te kodu države. Ova tablica služi kao šifrarnik te se na nju referenciraju tablice CA_DATA i ADDRESS.
- CERT_USER – služi za implementiranje korisnika certifikata. Jedini podatak u njoj je strani ključ kojim se ova tablica referencira na USER tablicu s kojom je u relaciji jedan naprema jedan.
- CERT – sprema podatke o korisničkim certifikatima poput serijskog broja, datuma kreiranja, datuma početka važenja certifikata, datuma isteka certifikata, te strane ključeve koji se referenciraju na tablice USER i USER_SMART_CARD. S tim relacijama je u vezi jedan naprema više.
- KEY_STORE – sprema korisničke ključeve i tip ključa. Referencira se na tablicu CERT stranim ključem i s tom relacijom je u vezi jedan naprema jedan.
- USER_SMART_CARD – sprema podatke o izdanim pametnim karticama. Sprema datum izdavanja, datum do kojeg kartica vrijedi te oznaku da li je kartica izgubljena.
- CRL – služi za spremanje podataka o opozvanim certifikatima. Sprema se serijski broj certifikata te datum opoziva. Povezana je stranim ključem s tablicom CRL_REASON_CODE.
- CRL_REASON_CODE – šifrarnik, sprema razloge opoziva certifikata po pravilima koje određuje X.509 norma.
- APP_USER – služi za implementaciju korisnika sustava. Sprema korisničko ime, lozinku te se referencira na APP_ROLE stranim ključem. Relacija između tih tablica je više naprema jedan. Također se referencira na tablicu USER u relaciji jedan naprema jedan.
- APP_ROLE – šifrarnik koji sprema autorizacijske nivoe. Prema podacima iz te tablice implementira se autorizacija PKI sustava.

7.2.3. Tehnologije korištene kod implementacije PKI sustava

Kod implementiranja PKI sustava korištena je Java Enterprise tehnologija koja omogućuje izradu web aplikacija, a kao pomoć kod izrade uporabljena je SpringFramework tehnologija koja olakšava i ubrzava izradu aplikacija, a također omogućuje jednostavnu nadogradnju sustava. Za komunikaciju s pametnom karticom koristilo se terminalsko sučelje opisano u poglavlju 6. Za kriptografiju se koristilo BouncyCastle programsko sučelje implementirano u Java programskom jeziku.

Kao poslužitelj na kojem se izvršava Java Enterprise aplikacija koristio se Apache Tomcat 5.0.28 poslužitelj. Dok je baza podataka realizirana u MySql sustavu.

8. Digitalna knjižnica u sustavu javnih ključeva

Digitalna knjižnica u sustavu javnih ključeva (PKI sustavu) demonstrira rad implementiranog programa (*appleta*) u Java Card tehnologiji (poglavlje 5.), implementiranog terminalskog sučelja (poglavlje 6.) te demonstrira korištenje PKI sustava (poglavlje 7.).

8.1. Sigurnosni zahtjevi digitalne knjižnice

Digitalna knjižnica je jedan od krajnjih korisnika ili entiteta implementiranog PKI sustava te ona posjeduje privatni i javni ključ koje koristi za kriptiranje podataka te digitalni certifikat kojim dokazuje svoj identitet.

Digitalna knjižnica mora omogućiti pristup samo registriranim korisnicima digitalne knjižnice koji su također registrirani u PKI sustavu. Kao autentifikacijski protokol kojim knjižnica dopušta pristup registriranim korisnicima koristi se X.509 protokol s tri poruke koji omogućuje obostranu autentifikaciju.

Uz autentifikaciju sustav digitalne knjižnice mora imati i sustav kontroliranja pristupa tj. autorizaciju nad podacima. Autorizacija se brine da autentificirani korisnik u digitalnu knjižnicu može preuzeti samo one dokumente za koje ima autorizacijska prava.

Sustav mora omogućiti spremanje i preuzimanje dokumenata digitalne knjižnice na taj način da može nepobitno dokazati da je baš ta osoba spremila ili preuzela dokument.

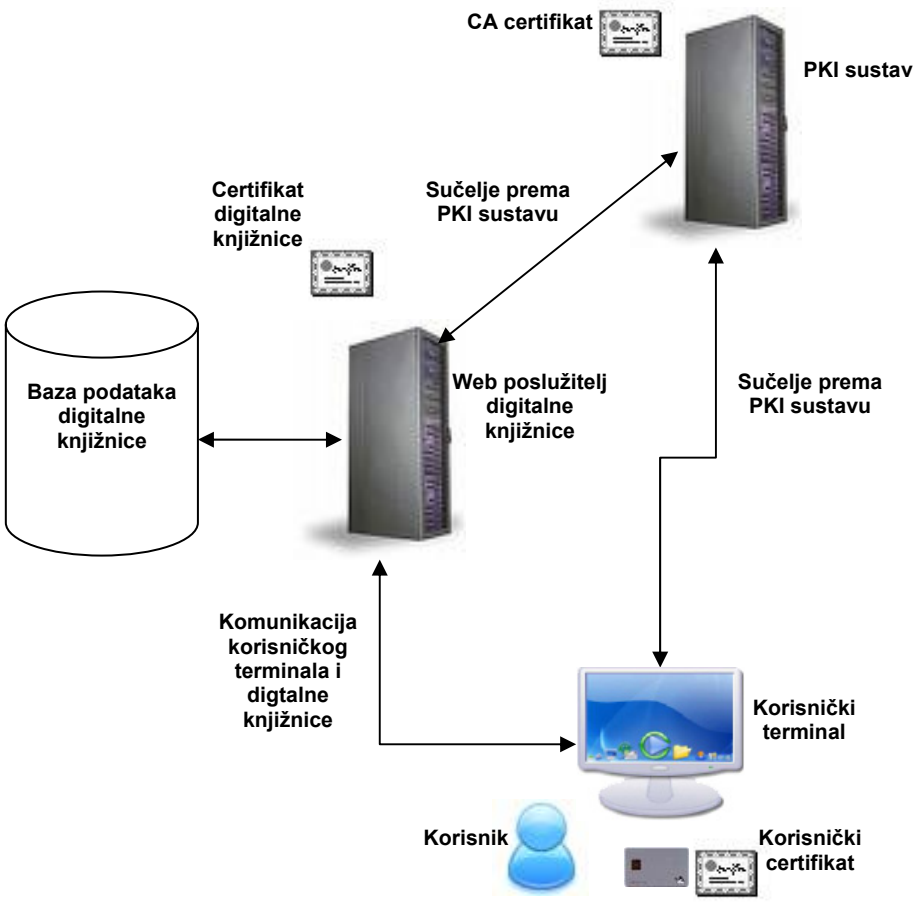
8.2. Arhitektura digitalne knjižnice

Digitalnu knjižnicu čini baza podataka digitalne knjižnice, web poslužitelj digitalne knjižnice te sučelja prema korisniku i PKI sustavu. Slika 8.1. prikazuje digitalnu knjižnicu u sustavu javnih ključeva.

Digitalna knjižnica i korisnik koji koristi usluge digitalne knjižnice imaju svoje certifikate te ključeve izdane od strane sustava javnih ključeva. Kako bi korisnik mogao koristiti usluge digitalne knjižnice on se mora autentificirati. Protokol autentifikacije, kao što je navedeno u prethodnom poglavlju, je X.509 protokol s tri poruke koji omogućuje provjeru identiteta obje strane. Kako bi sustav digitalne knjižnice mogao provjeriti da li korisnik ima valjan certifikat, on koristi sučelje prema PKI-u koji mu omogućava izdavanje liste opozvanih certifikata (CRL) te mu omogućava provjeru certifikata. Korisnički terminal preko kojeg se korisnik autentificira u sustav ima sučelje prema PKI-u koji njemu omogućava provjeru certifikata digitalne knjižnice.

Sustav digitalne knjižnice koristi bazu podataka u koju sprema dokumente, digitalne potpise dokumenata, narudžbu za preuzimanje dokumenta te digitalni potpis narudžbe. Uz te podatke sprema i podatke o registriranim korisnicima PKI sustava te njihova autorizacijska prava. Dakle, kako bi korisnik mogao koristiti sustav digitalne knjižnice on mora biti

registriran u digitalnu knjižnicu tj. nije dovoljno da je samo korisnik PKI sustava.

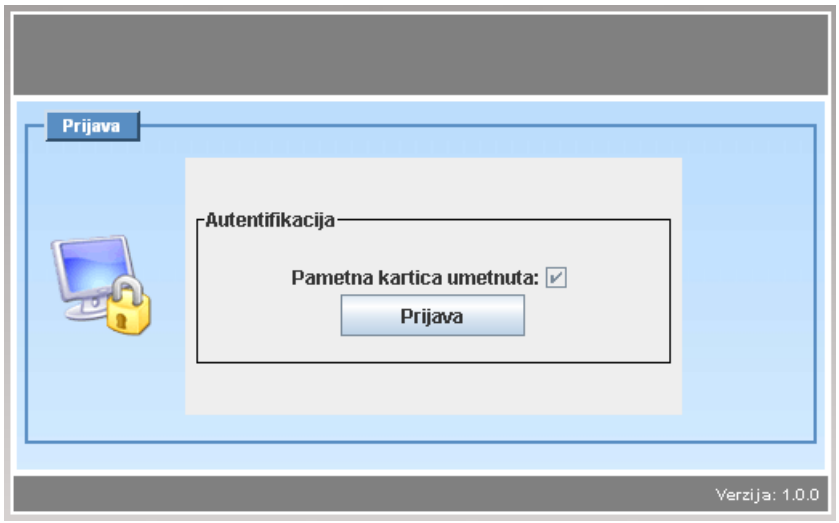


Slika 8.1. Digitalna knjižnica u sustavu javnih ključeva

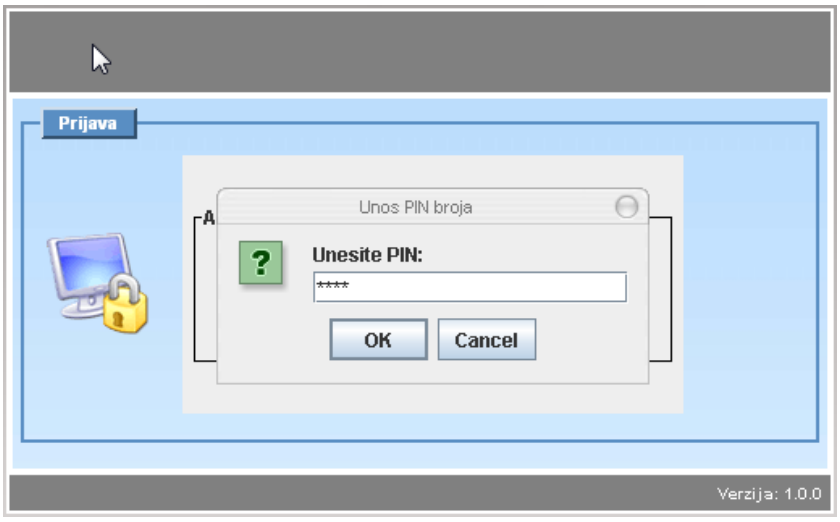
8.3. Autentifikacija i autorizacija korisnika

Za korištenje funkcija digitalne knjižnice korisnik se prvo mora autentificirati. Za autentificiranje korisnik mora imati pametnu karticu izdanu od PKI sustava koja sadržava njegove ključeve te certifikat. Međutim, u slučaju da korisnik nema pametnu karticu moguća je i autentifikacija s datotekom koja sadrži privatni ključ po PKCS#12 formatu. Za autentificiranje korisnika u sustav zadužen je Java *applet* kojeg pokreće web preglednik. *Applet* koji omogućuje autentificiranje nalazi se u jar (eng. *Java Archive*) arhivi *DigitalLibraryApplet.jar*. Ta arhiva je digitalno potpisana od strane poslužitelja digitalne knjižnice, te kako bi korisnik mogao koristiti zadani applet on mora prihvatiti certifikat poslužitelja. Slika 8.2. prikazuje početni ekran za autentificiranje u sustav digitalne knjižnice. Ukoliko je pametna kartica umetnuta u čitač, applet prikazuje poruku o tome i tada je moguća autentifikacija s pametnom karticom. U protivnom se autentifikacija obavlja datotekom koja sadrži ključ po PKCS#12 formatu. Slika 8.3. prikazuje ekran

za unos PIN broja kod autentifikacije s pametnom karticom, dok ekran 8.4. prikazuje početni ekran nakon uspješnog autentificiranja. Zaglavlje svakog ekrana nakon uspješne autentifikacije sadrži informaciju o trenutno autentificiranom korisniku te njegovoj razini pristupa.



8.2. Početni ekran autentifikacije

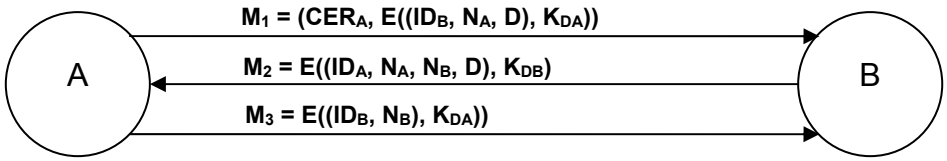


Slika 8.3. Unos PIN broja kod autentifikacije s pametnom karticom

Slika 8.4. Početni ekran nakon uspješne autentifikacije

8.3.1. X.509 autentifikacijski protokol s tri poruke

Protokol za autentifikaciju korisnika u digitalnu knjižnicu je X.509 protokol s tri poruke koji omogućuje obostranu autentifikaciju sudionika. Slika 8.5. prikazuje komunikaciju kod X509 autentifikacijskog protokola s tri poruke.



Slika 8.5. X.509 autentifikacijski protokol s tri poruke

Kad sudionik *A* želi komunicirati s sudinikom *B* on tada:

- generira nasumični broj N_B
- pronalazi put od *A* do *B* te iz certifikata CER_B saznaje i utvrđuje javni ključ KE_B osobe *B*
- oblikuje trojku (ID_B, N_A, D) gdje je *D* podatkovana komponenta koja može biti kriptirana javnim ključem KE_B
- kriptira trojku svojim privatnim ključem KD_A i šalje sudiniku *B* poruku:

$$M_1 = (CER_A, E((ID_B, N_A, D), K_{DA}))$$

Kada sudionik B primi poruku M_1 on:

- pronalazi u tablicama put od *B* do *A* te iz certifikata CER_A saznaje i utvrđuje javni ključ KE_A sudionika *A*
- s pomoću ključa KE_A dobiva trojku

$$(ID_B, N_A, D) = D((E((ID_B, N_A, D), K_{DA})), KE_A)$$

- na temelju ID_B utvrđuje da je poruka stvarno upućena njemu
- dekriptira svojim privatnim ključem KD_B podatkovnu komponentu *D* te po želji može usporediti nasumični broj N_A s brojevima iz prethodnih poruka da utvrdi da poruka nije ponovljena

Nakon toga sudionik *B*:

- generira svoj nasumični broj N_B
- oblikuje četvorku (ID_A, N_A, N_B, D), gdje je *D* podatkovna komponenta koja može biti kriptirana ključem KE_A
- kriptira oblikovanu četvorku ključem KD_B te šalje sudioniki *A* poruku:

$$M_2 = E((ID_A, N_A, N_B, D), K_{DB})$$

Kada sudinik *A* primi poruku M_2 on:

- pomoću ključa KEB dekriptira poruku M_2 te dobiva

$$(ID_A, N_A, N_B, D) = D((E((ID_A, N_A, N_B, D), K_{DB}), K_{EB})$$

- na temelju ID_A utvrđuje da je poruka stvaro upućena njemu
- dekriptira svojim privatnim ključem KD_A podatkovnu komponentu *D*
- po želji može provjeriti nasumični broj N_B s brojevima iz prethodnih poruka kako bi ustanovio da poruka nije ponovljena

Nakon toga sudionik *A*:

- sačinjava par (ID_B, N_B)
- kriptira dobiveni par ključem KD_A te šalje sudioniku B poruku:

$$M_3 = E((ID_B, N_B), K_{DA}))$$

Po primitku poruke M_3 sudionik *B*:

- pomoću ključa KE_A dekriptira poruku M_3 i dobiva par

$$(ID_B, N_B) = D((E((ID_B, N_B), K_{DA}))), KE_A)$$

- uspoređuje dobiveni NB iz poruke s izvornom vrijednošću te pomoću IDB utvrđuje da je poruka stvarno poslana njemu

8.3.2. Implementacija autentifikacijskog protokola

U sustavu digitalne knjižnice korisnik i sustav komuniciraju preko http protokola i tako razmjenjuju poruke. Za svaku od strana napravljena je implementacija koja generira, obrađuje i šalje poruke. Poruke kojima se komunicira jednake su porukama iz poglavlja 8.3.1. uz iznimku da poruke M_1 i M_2 ne koriste podatkovnu komponentu. Kao identifikator sudionika koristi se serijski broj certifikata pojedinog sudionika jer je taj broj jedinstven u sustavu javnih ključeva te je po njemu moguće jednoznačno odrediti vlasnika certifikata.

Na terminalskoj strani autentifikaciju implementira Java *applet*. U njemu je implementirano sučelje `AuthenticationService` koje sadrži metode za generiranje prve i treće poruke te provjeru druge poruke. Implementacija ovog sučelja koristi sučelje `AuthCryptoService` koje ima funkciju kriptiranja, dekriptiranja te dobavljanja korisničkog i poslužiteljskog certifikata. Ovo sučelje implementiraju dva razreda ovisno o tome da li se autentifikacija obavlja pametnom karticom ili datotekom sa spremljenim ključem u PKCS#12 formatu. Ti razredi su `ScAuthCryptoService` i `P12AuthCryptoService`. Certitifikat digitalne knjižnice nalazi se u *jar* arhivi (*DigitalLibraryApplet.jar*) u kojoj se nalazi applet za autentificiranje. Prije

početka same autentifikacije taj certifikat se provjerava te ako je važeći kreće se s autentifikacijom.

Na poslužiteljskoj strani servis autentifikacije implementiran je pomoću Java Enterprise Filter tehnologije. Filter je ulazna točka u web aplikaciju temeljena na Java Enterprise tehnologiji. Za svaki filter se određuje internet dio adrese na koju se on pokreće. Taj dio nalazi se iza imena cijele web aplikacije. U slučaju digitalne knjižnice to je "*server:port/DigitalLibrary/*". Kako bi se autentifikacijski filter za aplikaciju pokrenuo potrebno je iz terminala pozivati adresu koja u sebi sadrži *"j_security"*. Nakon što se filter pokrene on pročita podatke iz zahtjeva koje mu je poslao terminal. Nakon što filter primi poruku M_1 on obradi tu poruku, provjeri certifikat korisnika te provjeri da li je taj korisnik registriran u sustavu digitalne knjižnice i ako su zadovoljeni svi uvjeti šalje poruku M_2 terminalu. Nakon toga čeka poruku M_3, te nakon što dobije poruku provjerava je i ako su uvjeti provjere zadovoljeni šalje terminalu odgovor da se preusmjeri na početnu stranicu digitalne knjižnice, sprema podatke korisnika u sesiju i čeka na sljedeće zahtjeve.

8.3.3. Implementacija autorizacije

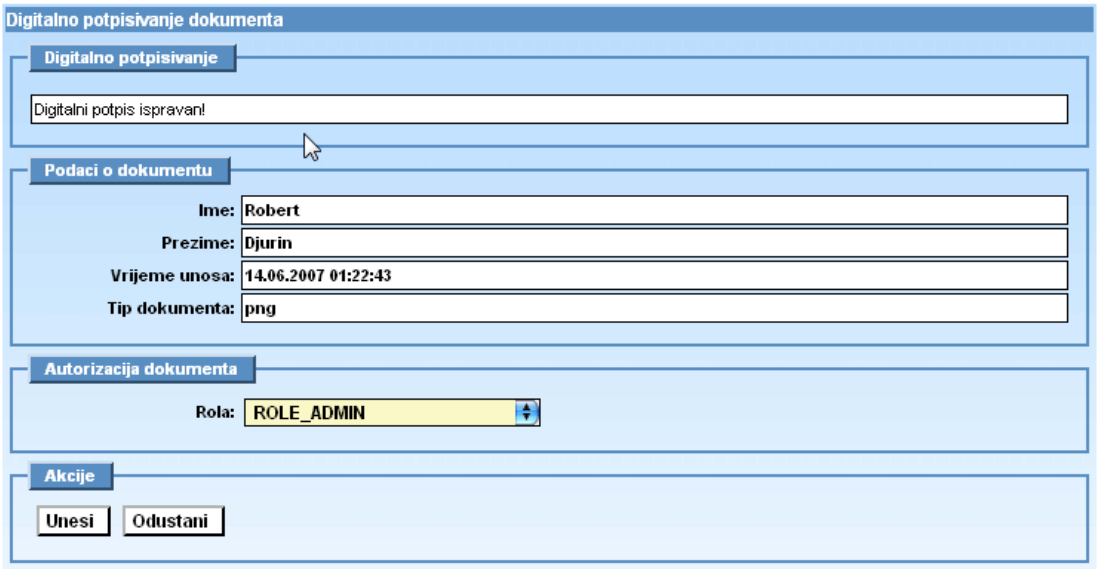
Sustav digitalne knjižnice ima mogućnost izdavanja dokumenata korisnicima koji imaju autorizacijska prava na te dokumente. Također je omogućeno da samo administratori sustava digitalne knjižnice mogu registrirati nove korisnike. Dakle, na neke stranice pristup imaju samo administratori. Sustav digitalne knjižnice ima dvije razine autorizacije: administrator i korisnik. Implementacija autorizacije obavljanja određenih funkcija riješena je kao i autentifikacija preko filtera. Međutim u ovom slučaju filter se pokreće na bilo koju adresu. Filter je napravljen tako da ga je moguće konfigurirati. Slika 8.6. prikazuje konfiguraciju filtera u SpringFramework tehnologiji u *xml* formatu.

Filter sadrži nekoliko vrijednosti. Prva vrijednost je `loginPage` i njezina vrijednost se koristi za preusmjeravanje na ekran za autentifikaciju ako se pokuša doći do resursa za kojeg ne postoji dozvola ili ako korisnik nije autentificiran. Druga vrijednost je `anonymousPages` koja predstavlja listu dijelova adrese aplikacije kojima smiju svi pristupiti. To su u ovom slučaju ekran za autentifikaciju, ekran koji se pojavljuje nakon odjave, direktorij slika, direktorij *jar* arhiva za *applete* te direktorij dekoratora stranica. Nakon toga slijedi vrijednost `defaultPages` koja sadrži listu dijelova adrese kojima smiju pristupiti svi autentificirani korisnici. Na kraju postoji vrijednost `rolePages` implementirana kao mapa koja za ključeve ima autorizacijske vrijednosti pojedinih stupnjeva autorizacije ili rola. Tako se iz slike 8.6. vidi da administrator ima dozvolu pristupa stranicama za administraciju korisnika te pregleda akcija dok korisnici bez administratorskih ovlasti to nemaju.

Kod autorizacije dokumenata za svaki se dokument prije spremanja odabire autorizacijski stupanj (slika 8.7.) koji korisnik mora imati kako bi mogao preuzeti taj dokument. Korisnici s administratorskim pravima mogu preuzimati sve dokumente dok obični korisnici mogu preuzimati samo dokumente koji su njima dostupni.

```
<bean id="checkAuthorizationFilter"
    class="hr.rd.diglib.authentication.filter.CheckAuthorizationFilter">
    <property name="loginPage" value="login.html" />
    <property name="anoymousPages">
        <list>
            <value>j_security</value>
            <value>css/*</value>
            <value>images/*</value>
            <value>appletAuthentication/*</value>
            <value>applets/*</value>
            <value>login.html</value>
            <value>logoff.html</value>
        </list>
    </property>
    <property name="defaultPages">
        <list>
            <value>index.html</value>
            <value>document/*</value>
            <value>appletSign/*</value>
        </list>
    </property>
    <property name="rolePages">
        <map>
            <entry key="ROLE_ADMIN">
                <list>
                    <value>users/*</value>
                    <value>actions/*</value>
                </list>
            </entry>
        </map>
    </property>
</bean>
```

Slika 8.6. Konfiguracija filtera za autorizaciju

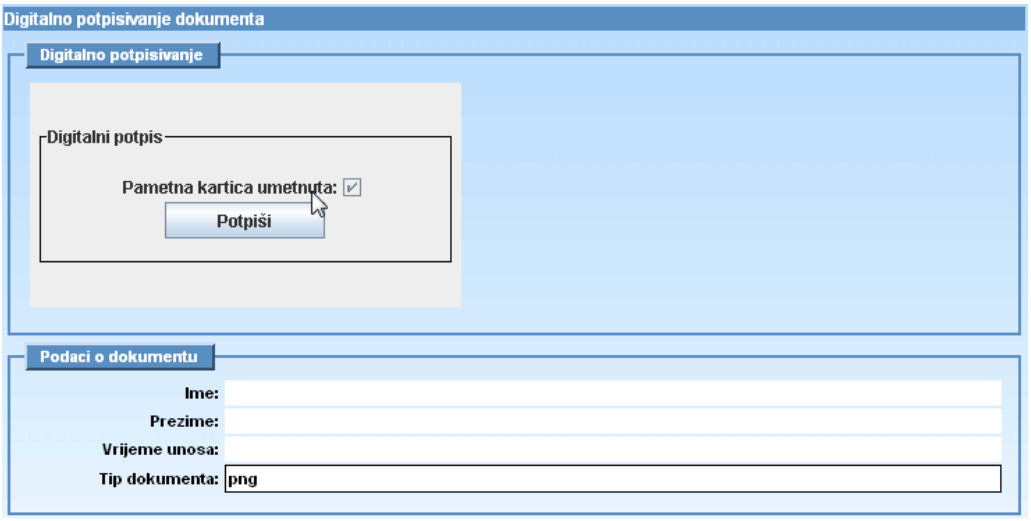


Slika 8.7. Spremanje dokumenta

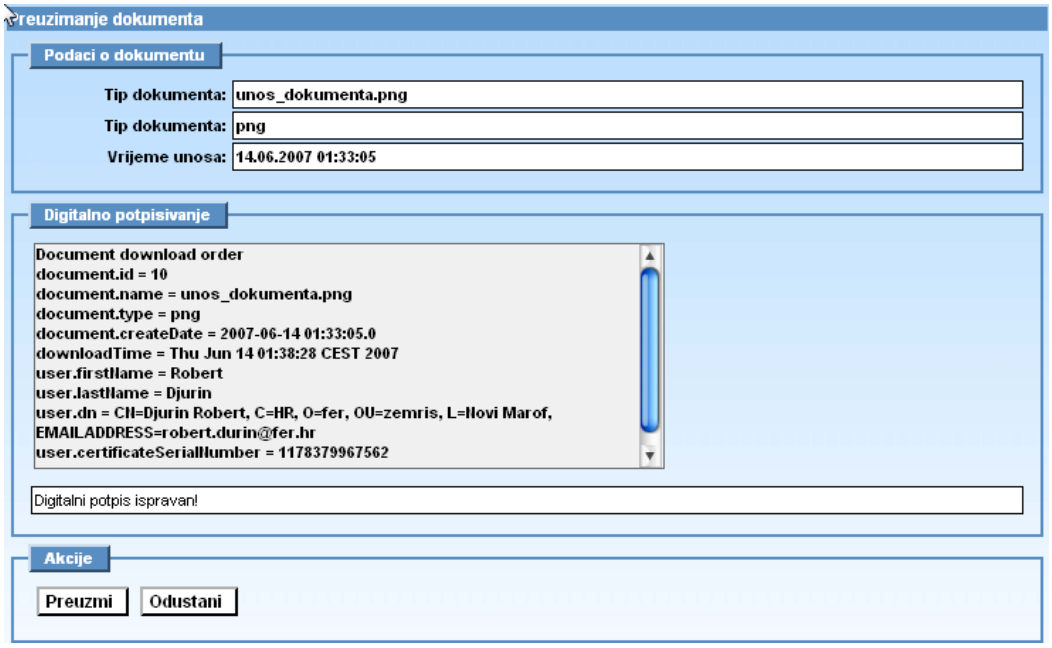
8.4. Rad s dokumentima

Sustav digitalne knjižnice omogućava korisnicima pohranu i preuzimanje dokumenata. Prije same pohrane i preuzimanja dokumenta korisnik mora prvo potpisati taj dokument (slika 8.8.), odnosno narudžbu kod preuzimanja dokumenta (slika 8.10). Nakon što korisnik napravi digitalni potpis sustav digitalne knjižnice provjera taj potpis i ako je potpis ispravan dopušta se spremanje (slika 8.7.) ili preuzimanje dokumenta (slika 8.9.).

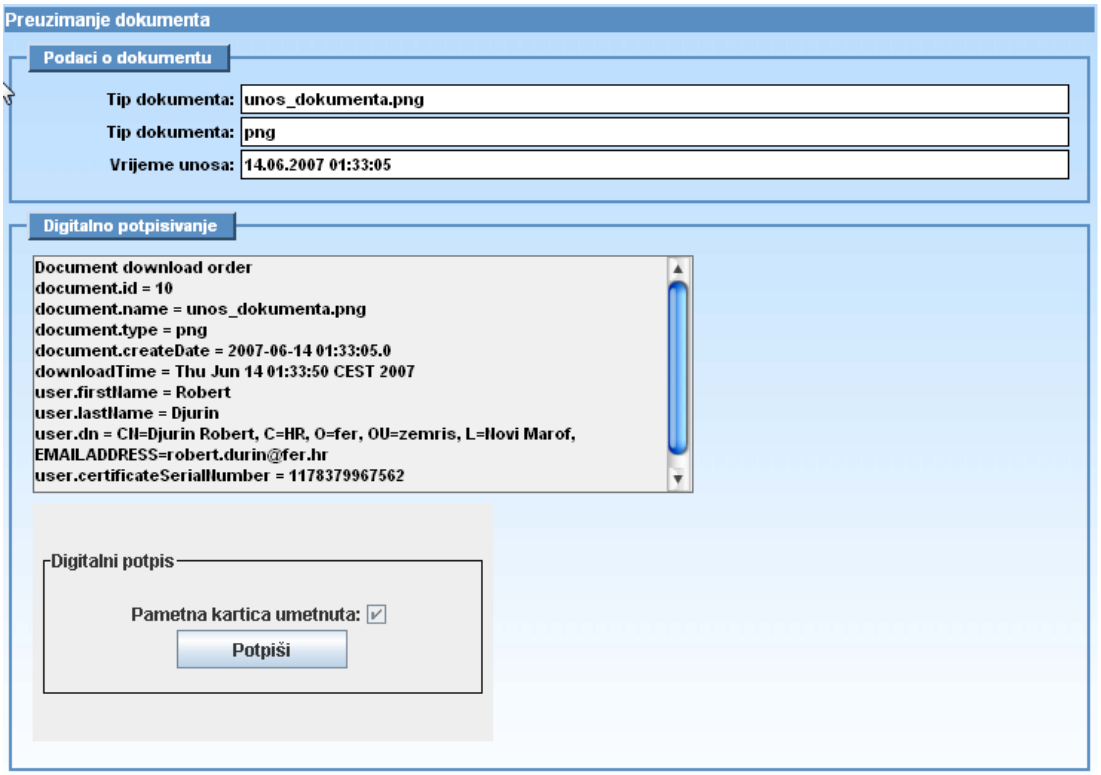
Algoritam koji se koristi za digitalno potpisivanje dokumenata i narudžbi je SHA1-RSA. Prilikom kreiranja digitalnog potpisa pametnom karticom klijent radi sažetak dokumenta ili narudžbe te ga šalje pametnoj kartici na potpisivanje.



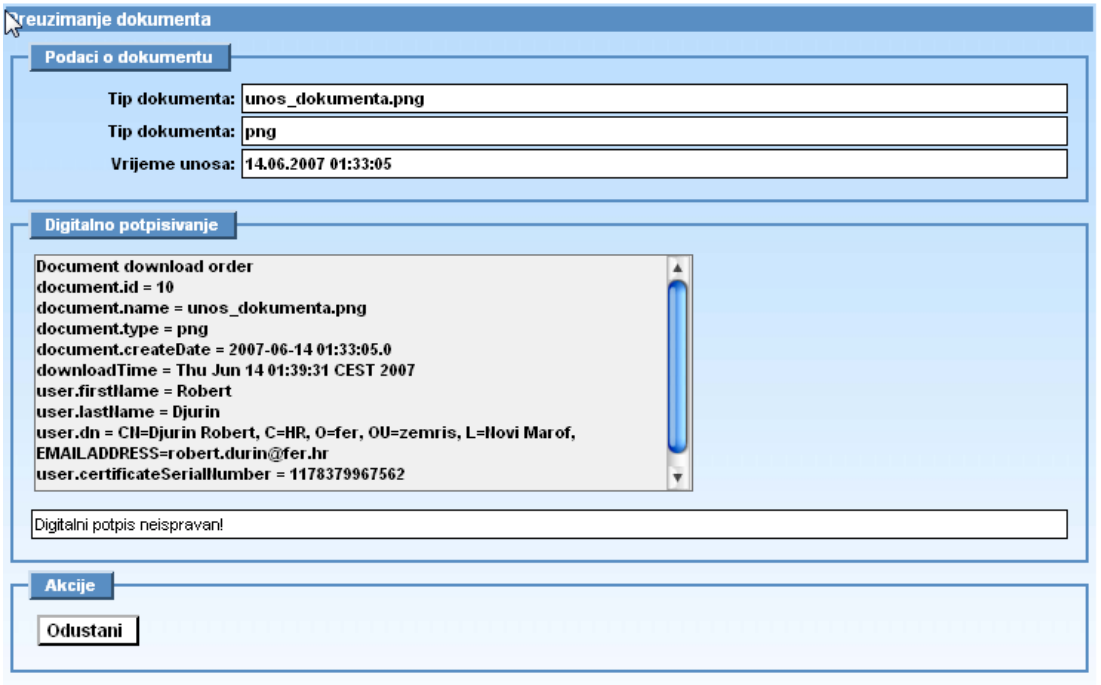
Slika 8.8. Digitalno potpisivanje dokumenta



Slika 8.9. Preuzimanje dokumenta ako je potpis ispravan



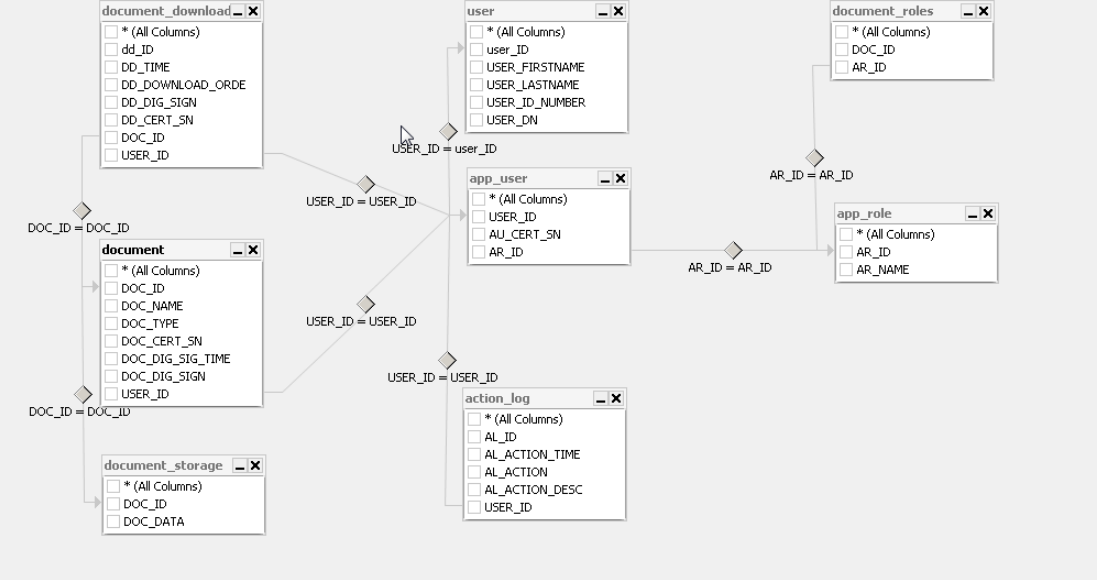
Slika 8.10. Digitalno potpisivanje narudžbe za preuzimanje dokumenta



Slika 8.11. Nemogućnost preuzimanja dokumenta ako je potpis neispravan

8.5. Baza podataka digitalne knjižnice

Baza podataka digitalne knjižnice implementirana je u MySql sustavu. Ona služi za spremanje dokumenata, narudžbi, digitalnih potpisa te podataka o korisnicima sustava i akcijama koje su korisnici radili u sustavu. Baza podataka digitalne knjižnice sastoji se od osam tablica. Model baze podataka za digitalnu knjižnicu prikazan je na slici 8.12.



Slika 8.12. Model baze podataka digitalne knjižnice

- USER – sprema podatke o registriranim korisnicima u sustav kao što su ime, prezime, identifikacijski broj te DN korisnika.
- APP_USER – implementira korisnika sustava, sprema podatke o serijskom broju certifikata tog korisnika te se referencira na relaciju USER i APP_ROLE
- APP_ROLE – sifrarnik autorizacijskih prava korisnika te dokumenata.
- DOCUMENT – sprema podatke o spremljenom dokumentu kao što su ime dokumenta, tip dokumenta, serijski broj certifikata kojim je dokument potpisan, vrijeme digitalnog potpisa, te sam digitalni potpis. Referencira se na tablu APP_USER preko čega se dobivaju informacije o korisniku koji je spremio dokument.
- DOCUMENT_STORAGE – sprema binarno dokument u bazu, referencira se na tablu DOCUMENT.
- DOCUMENT_DOWNLOAD – sprema podatke o izdanim dokumentima te narudžbu za izdavanje i njen potpis. Referencira se na tablu APP_USER kojom se određuje korisnik koji je preuzeo dokument.

- DOCUMENT_ROLES – veza između table DOCUMENT i APP_ROLE čime se dobije relacija više naprama više.
- ACTION_LOG – služi za spremanje akcija koje su korisnici sustava radili tijekom korištenja sustava digitalne knjižnice.

8.6. Zaključna razmatranja

Aplikacija digitalne knjižnice demonstrira korištenje pametne katice u sustavu javnih ključeva za autentifikaciju i digitalno potpisivanje. Sustav digitalne knjižnice omogućuje spremanje dokumenata u digitalnom obliku te preuzimanja dokumenata iz digitalne knjižnice. Za svaki dokument može se neporecivo utvrditi korisnik koji je taj dokument spremio ili preuzeo iz digitalne knjižnice. Korisnik koji sprema ili preuzima dokument iz digitalne knjižnice mora prije samih akcija spremanja i preuzimanja digitalno potpisati dokument u slučaju spremanja ili narudžbu u slučaju preuzimanja. Dokumenti u digitalnoj knjižnici imaju autorizacijski atribut koji označava stupanj autorizacije korisnika koji može preuzeti taj dokument. Za akcije autentificiranja, spremanja te preuzimanja dokumenata vodi se evidencija o akciji te korisniku koji obavlja tu akciju. Korisnici koji imaju administratorski stupanj autorizacije mogu pregledavati evidenciju akcija, a također mogu i dodavati nove korisnike u sustav.

Pametna kartica u sustavu digitalne knjižnice olakšava autentifikaciju korisnika te digitalno potpisivanje dokumenata. Za autentifkaciju u sustav korisnik treba samo staviti karticu u čitač te upisati ispravan PIN, a istu akciju treba obaviti i kod digitalnog potpisivanja dokumenta ili narudžbe.

Osnovna funkcionalnost implementirana je u sustav digitalne knjižnice i sustav je u potpunosti funkcionalan za upotrebu. Uz osnovnu funkcionalnost autentificiranja u sustav, digitalno potpisivanje dokumenata te spremanje i preuzimanje dokumenata implemetirana je i funkcionalnost registracije korisnika u sustav digitalne knjižnice te evidencija akcija. U sustav nije implementirana mogućnost brisanja pojedinog korisnika iz sustava te je za tu akciju potrebno korisnika obrisati iz baze podataka. Također nije implementirano pretraživanje preuzetih dokumenata. Kako bi sustav olakšao administraciju korisnika i evidenciju preuzetih dokumenata potrebno je implementirati ove funkcije u sustav digitalne knjižnice.

9. Zaključak

U ovom radu prikazana je implementacija programa za pametnu karticu u Java Card tehnologiji koja uz osnovno poznavanje Java programskog jezika omogućuje brzu i efikasnu izgradnju programa za pametnu karticu. Uz program za pametnu karticu implementirano je i sučelje koje se koristi za komunikaciju s pametnom karticom. Za demonstriranje rada pametne kartice implementiran je sustav javnih ključeva te sustav digitalne knjižnice. Pametna kartica koristila se za sigurnu autentifikaciju korištenjem X.509 protokola s tri poruke u sustav digitalne knjižnice, kriptiranje i dekriptiranje poruka te za spremanje podataka na nju. U ovom slučaju to su bili digitalni certifikati, međutim uz njih se mogu spremati i drugi podaci te informacije potrebne za rad nekog drugog sustava od ovdje implementiranog. Cijela implementacija diplomskog zadatka napravljena je besplatnim alatom i programskim sučeljima u programskom jeziku Java.

Pametne kartice danas polako postaju stvar ili bolje rečeno uređaj kojeg kojeg ćemo svi imati u novčaniku. Do sve veće popularnosti pametnih kartica dolazi zbog njezine glavne prednosti naspram magnetskih i memorijskih kartica, a to je sigurnost koju pametna kartica pruža za podatke spremljene na njoj. Java Card tehnologija jedna je od vodećih tehnologija za izradu programa namijenjenih izvršavanju na pametnoj kartici. Jedini nedostatak zasad je da nema besplatnog razvojnog alata za Java Card tehnologiju koji bi omogućio brzo ispravljanje grešaka u programima (eng. *debuging*). Međutim bez obzira na ovaj nedostatak Java Card tehnologiju čeka svijetla budućnost.

10. Literatura

[1] IBM Redbooks, *Smart Cards: A Case Study*, dostupno na Internet adresi *http://www.redbooks.ibm.com/redbooks/pdfs/sg24539*

[2] Sun Microsystems, *Java Card™ 2.1.1 Virtual Machine Specification,* 2000.

[3] Sun Microsystems, *Java Card™ 2.1.1 Runtime Environment (JCRE) Specification, 2000.*

[4] Sun Microsytems, *Java Card™ Development Kit User's Guide v2.2.1.*

[5] Schlumberger, *Cyberflex® Access™ Cards Programmer's Guide*

[6] Schlumberger, *Cryptoflex™ Card's Programmer's Guide*

[7] OpenCard consortium, *http://www.opencard.org*

[8] CardWerk Technologies, *http://www.cardwerk.com*

[9] Đurin R., *Višenamjenska pametna kartica*, seminar Zagreb 2005.

[10] Rebac F., Digitalna knjižnica u sustavu javnih ključeva, diplomski rad Zagreb 2006.

[11] Budin L., *Predavanja iz predmeta Operacijski sustavi 2*, Fakultet elektrotehnike i računarstva Sveučilišta u Zagrebu, Zagreb 2003.

[12] Rebac F., *Infrastruktura javnog ključa*, Seminarski rad, Fakultet elektrotehnike i računarstva Sveučilišta u Zagrebu, Zagreb 2005.

[13] Interface 21, *The Spring Framework – Reference Documentation*, dostupno na Internet adresi *http://static.springframework.org/spring/docs/2.0.x/reference/index.htm*

[14] RSA Laboratories, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) profile,* dostupno na Internet adresi *http://tools.ietf.org/html/rfc3280#section-5.3.1*

[15] The Apache software foundation, *Tomcat 5.0.28. manual* dostupno na Internet adresi *http://tomcat.apache.org*