ELEKTRONIČKO POSLOVANJE
SKRIPTA ZA 2. KOLOKVIJ (cijela godina)

1. ARHITEKTURA ELEKTRONIČKOG POSLOVANJA

-podlogu/osnovu infrastrukture el.poslovanja predstavlja arhitektura IS-a.

Elektroničko poslovanje – sustav izvršavanja svih elemenata poslovnih aktivnosti elektroničkih putem. Oslanja se na upotrebu ICT-a odnosno sve poslove i procese održava i realizira upotrebom ICT-a. (npr. kupujemo digitalni sadržaj, a ne artikl kao u dućanu)

Prednosti elektroničkog poslovanja:

- poslovanje bez papira (elektronički dokumenti zamjenjuju papirne dokumente)
- poslovanje bez ureda (mogućnost rada od kuće)
- niži troškovi poslovanja
- brža komunikacija odnosno razmjena informacija
- „zeleniji“ tip poslovanja (ne treba se trošit papir, boja, prijevoz itd.)

Informacijski sustav – dio poslovnog sustava koji prihvaća, obrađuje, sprema informacije i pruža ih drugima

Arhitektura IS-a – prikazuje i opisuje kako su međusobno organizirani i povezani osnovni elementi IS-a te odnos između tih elemenata
-slojevi arhitekture IS-a:

- Arhitektura poslovnih procesa (na koji način su poslovni procesi vezani uz IS)
- Sistemska arhitektura (na koji način su hardverski elementi međusobno povezani)
- Tehnička arhitektura
- Arhitektura proizvodnog/uslužnog sustava (specijalizirana za kontekst proizvodnog i/ili uslužnog poduzeća

Višeslojna ili troslojna arhitektura IS-a
-najstariji oblik arhitekture IS koji je prisutan i danas
-elementi (hw, sw, ow, nw, dw) su organizirani u:

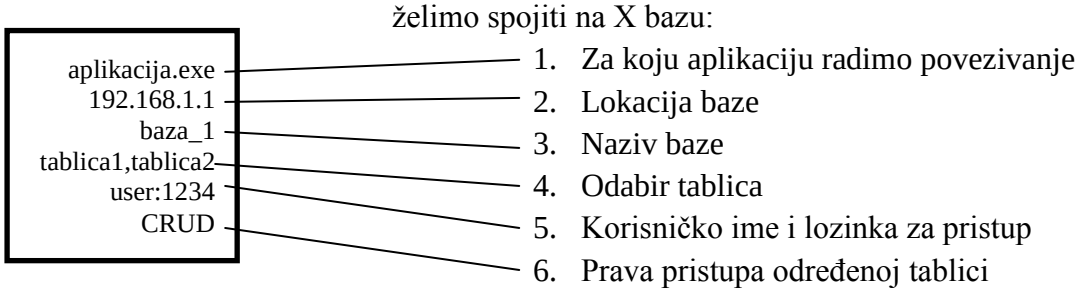
- Fizički sloj (vidljiv)
- Logički sloj

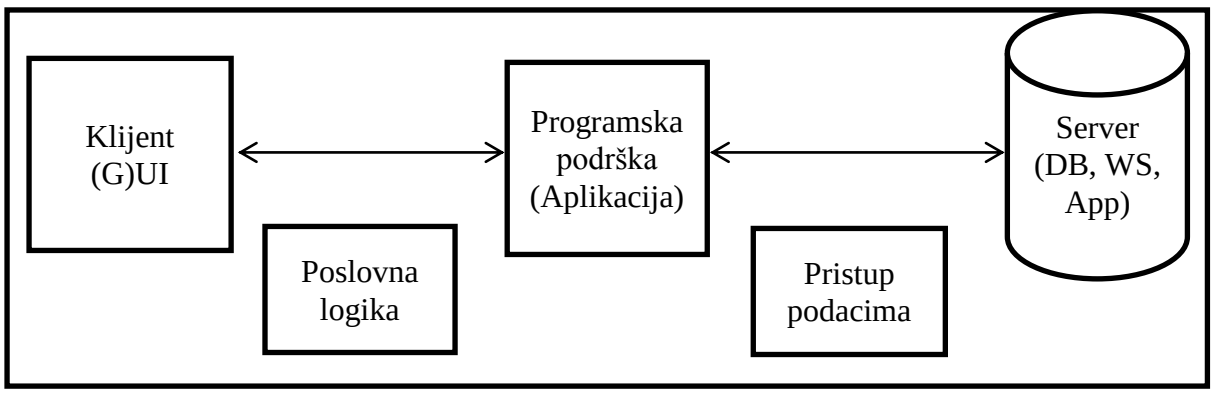
-temelji se na postojanju 3 sloja:

- Klijentska strana (računalni proces koji koristi IS, sadrži ono što se nalazi kod krajnjeg korisnika odnosno klijenta)
- Programska podrška (sučelje, aplikacije u IS-u, sadrži sve one aplikacije koje korisnik koristi, ima instalirane)
- Server (djeljivi resurs, računalo koje istovremeno može posluživati više računala te omogućuje korištenje jednog podatka od strane više korisnika – web server, e-mail servis, transakcijski server, aplikativni server itd.)

-uz gornje slojeve postoji još 2 međusloja:

- Međusloj poslovne logike (međusloj između klijenta i programske podrške, služi za podešavanje programske podrške odnosno podešavanje sustava radu poslovnog procesa; inače je dio sloja programske podrške ali se ponekad izdvaja kako bi se lakše upravljalo i parametriziralo IS)
- Međusloj pristupa podacima (radi vezu između aplikacije i odgovarajuće baze podataka i govori gdje se nalazi baza podataka koju povezujemo)
  ODBC (Open Database Connectivity) – windows inačica ovakvog sloja
  -Primjer: koraci koji se obave u ovom sloju kada se želimo spojiti na X bazu:





Slika 1 - klijent-server arhitektura

Pojavni oblici troslojne arhitekture IS-a

-gleda se strana na kojoj se nalazi programska podrška

-postoje tri pojavna oblika:

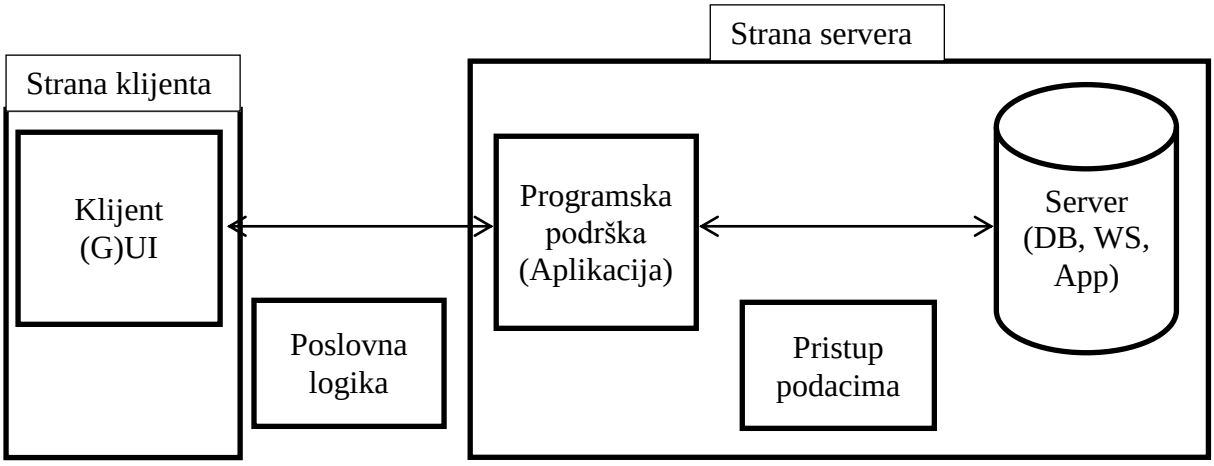
1) „tanki“ klijent – „debeli“ server

   -prvobitni oblik

   -karakterizira ga objedinjavanje sloja programske podrške i sloja servera na velika mainframe računala, klijent preko sučelja na klijentskoj strani daje naredbe i prima rezultate obrade

   -klijent predstavlja samo ulazno-izlaznu komponentu, dok server obavlja sve druge poslove

   -sve popularniji je danas uz pojavu web aplikacija, tableta, smarthphone itd.



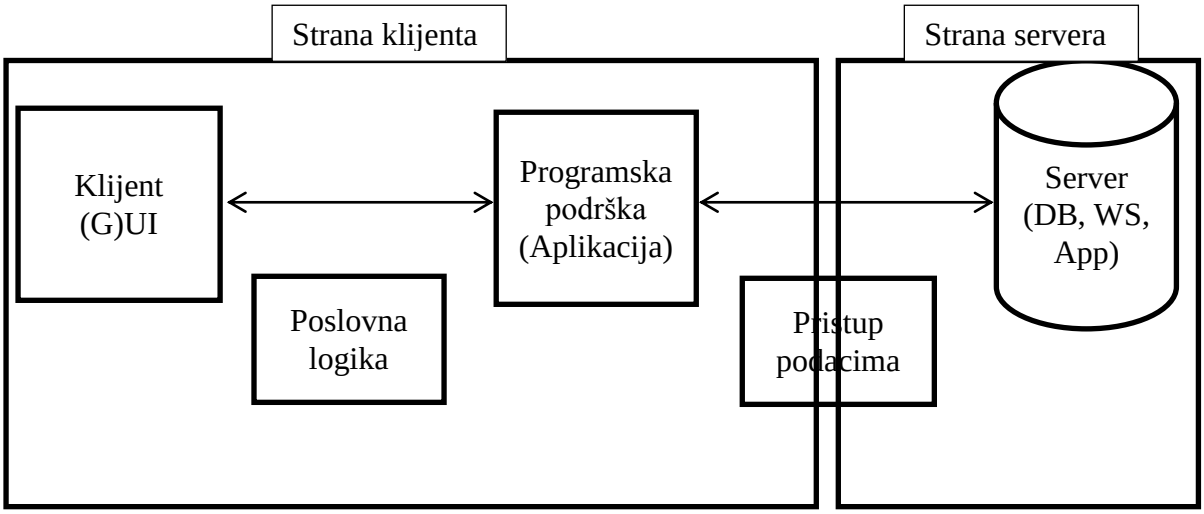
Slika 2 - tanki klijent - debeli server

2) „debeli“ klijent – „tanki“ server

   -nastao evolucijom „tanki klijent-debeli server“ oblika zbog pristupačnosti osobnih računala i ostale informacijsko-telekomunikacijske opreme

   -sloj programske podrške se seli sa strane servera na stranu klijenta (pr. Osobno računalo, sve što trebamo za normalnu upotrebu već je instalirano na računalo, nije potrebna povezanost sa internetom)

   -sloj servera služi za pohranu podataka, ali i bez tog sloja se može raditi



Slika 3 - debeli klijent - tanki server

3) client side - server side (CSSS)

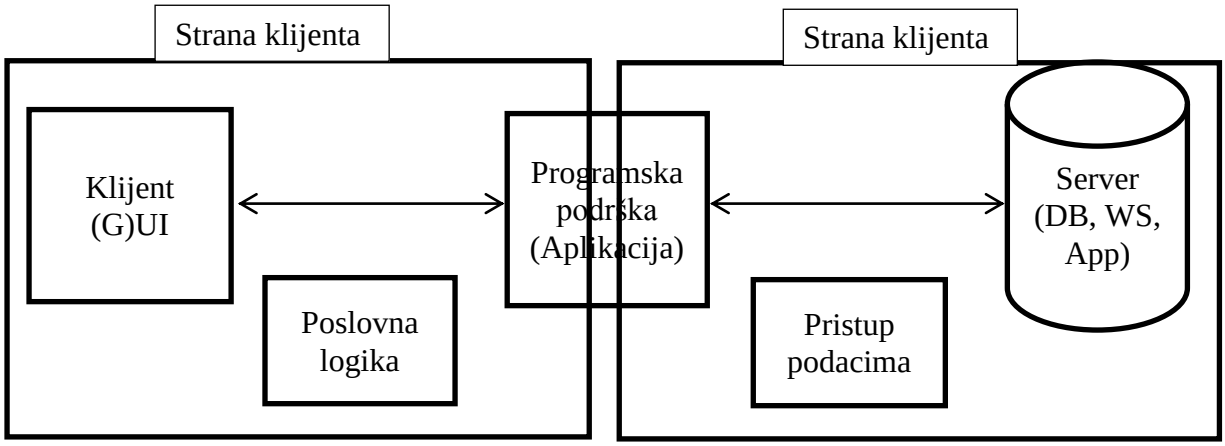
-kompromis između prijašnja dva pojavna oblika

-sloj programske podrške je podijeljen, neke aplikacije se nalaze na strani servera neke na strani klijenta - smještaj ovisi o tome gdje aplikacije bolje pripada odnosno radi

-npr. uredski paketi – instalirani na klijentskoj strani

-npr. moodle – instaliran na serverskoj strani

-najzastupljeniji oblik danas iako se danas zbog razvoja tehnologije (pojava web aplikacija, tablet računala, pametnih mobitela itd) vraća na prvobitni oblik arhitekture tankog klijenta i debelog servera - npr. na klijent strani (kod tableta, smartphone) imamo sučelje putem kojeg pristupamo aplikacijama i podacima koji su smješteni na serverskoj strani.



Slika 4 - client side - server side

Nove arhitekture IS-a:

ASP (Application Service Provider)

-omogućuje iznajmljivanje aplikacija koje se nalaze na serveru

-bazira se na mrežnoj povezanosti, bez nje nema ni aplikacije!

SAAS (Software As A Service)

-omogućuje korištenje aplikacije kada je to potrebno

IAAS (Infrastructure As A Service)

-omogućuje korištenje „tuđih“ harverskih resursa

-npr. AWS (Amazon Web Service) – omogućuje zakup web servisa odnosno računala

SOA (Servisno Orijentirana Arhitektura)

-ICT arhitektura koja pruža fleksibilnost potrebnu za implementiranje elemenata poslovnog procesa i postavljanje potrebne ICT infrastrukture u obliku sigurnih, standardiziranih komponenti koje se mogu višestruko koristiti i međusobno kombinirati kako bi zadovoljile različite poslovne potrebe

-postavlja temelje funkcioniranja Cloud-a

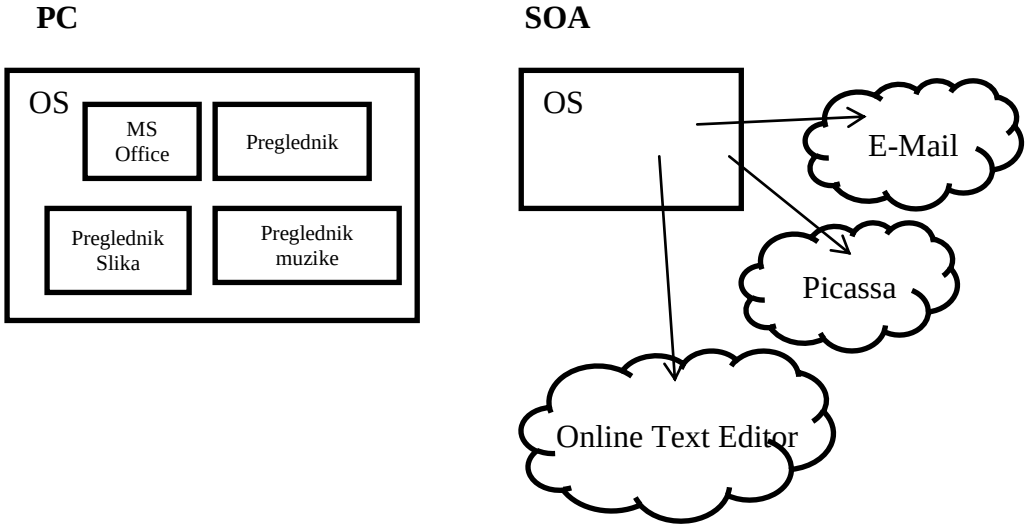
-arhitektura temeljena na servisima (web servis), oslanja se na servise koji su dostupni na računalnim mrežama
-servisi i korisnici su međusobno neovisni (bez povezanosti karakteristične za objekte)

-implementacija SOA-e:

- razvoj aplikacija koje koriste servise
- razvoj aplikacija koje se koriste kao servisi

-SOA se temelji na:

- labavoj povezanosti (način povezivanja programskih komponenti, komponente nisu ovisne jedna o drugoj)
- višestrukoj iskoristivosti (omogućuje da servis koristi više korisnika)
- interoperabilnosti (svojstvo više sustava da međusobno funkcioniraju (npr. zapis sa MUP-a može čitati FOI; gmail na androidu i iOS)



Slika 5 - odnos PC i SOA (pristup aplikacijama)

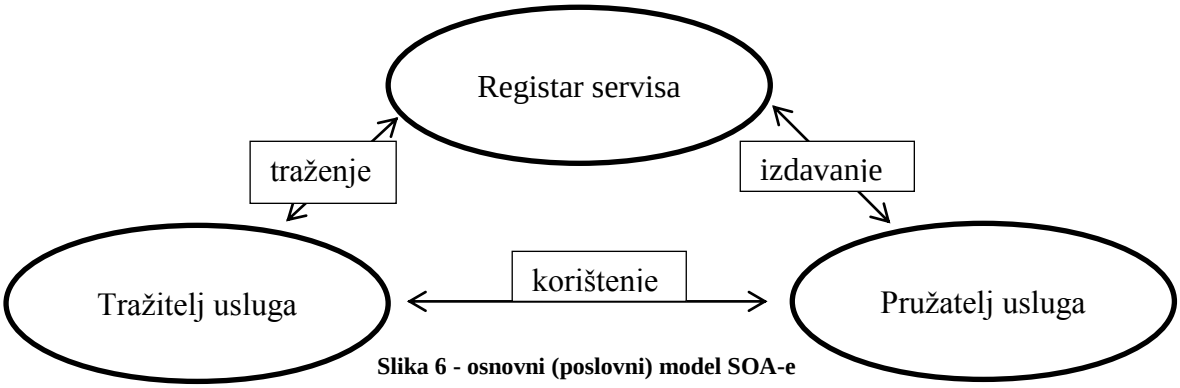
-ključ funkcionalnosti SOA je posjedovanje pristupnih sučelja – omogućuje interoperabilnost
WSDL (Web Service Definition Language)
-pristupno sučelje
-opis govori:

1) lokacija
2) pristupni podaci
3) ulazni parametri
4) rezultat obrade
5) funkcije koje imamo (npr. gmail – odgovori, obriši, otvori itd.)

-karakteristike SOA-e:

- interoperabilnost (svojstvo više sustava da međusobno funkcioniraju)
- raspoloživost (dostupnost 24/7)
- fleksibilnost (ne smije se narušavati funkcionalnost SOA-e prilikom dodavanja ili isključivanja pojedinih servisa)
- heterogenost (oblik interoperabilnosti, ali vezana uz legacy aplikacije)
- sigurno upravljanje komponentama (privatnost podataka, posl.procesa)

-osnovni (poslovni) model SOA-e:



Slika 6 - osnovni (poslovni) model SOA-e

-Tražitelj usluga – „mi“, tvrtke koje žele izgraditi SOA, traže odgovarajuće web servise
-Registar servisa – mjesto „sastanka“ tražitelja i pružatelja usluga, web mjesto putem kojeg tražitelj usluga traži potrebni servis odnosno web mjesto putem kojeg pružatelj izdaje svoj servis
-Pružatelj usluga – tvrtke koje su razvile servise i te servise nude tržištu

-SOA – tehnički model

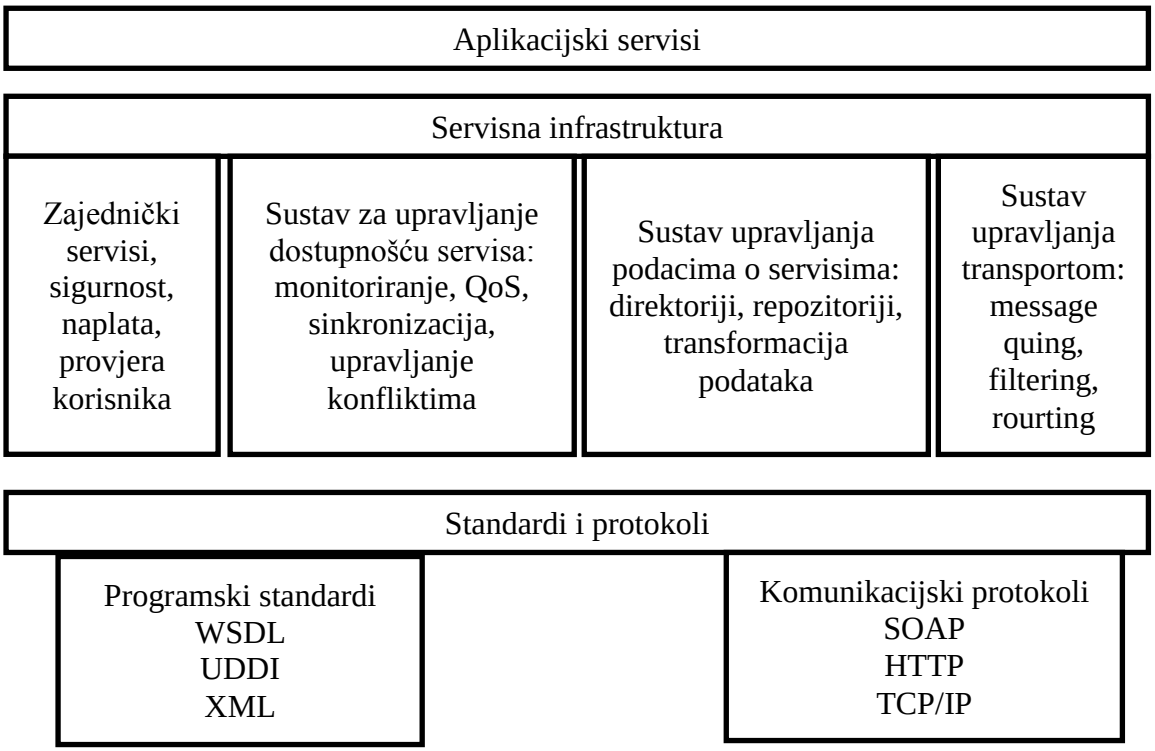
<table>
<tr><td>5.sloj pristupa/prezentacija</td><td rowspan="5">6. integracija</td><td rowspan="5">7.osiguranje kvalitete usluga, sigurnost, upravljanje, nadzor</td></tr>
<tr><td>4.sloj koreografije</td></tr>
<tr><td>3.sloj servisa</td></tr>
<tr><td>2.sloj komponenti</td></tr>
<tr><td>1.sloj operacijskog sustava</td></tr>
</table>

Slika 7 - tehnički model SOA-e

1) Sloj operacijskog sustava – nema veze sa Windowsima, MacOS, Linux, već se OS-om smatraju legacy aplikacije, aplikacije koje već postoje u našem IS i koje preuzimamo i na temelju kojih gradimo SOA
2) Sloj komponenti – obuhvaća poslovne procese koje želimo podržati i informatizirati unutar SOA, odabiremo procese za koje ćemo u 3.sloju razviti, kupiti ili unajmiti odgovarajuće web servise
3) Sloj servisa – razvijamo, kupujemo i unajmljujemo potrebne web servise

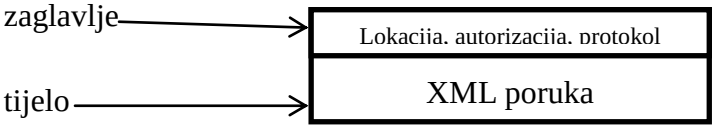
4) Sloj koreografije – povezivanje web servisa u smislenu cjelinu koja će predstaviti i omogućiti funkcionalnost IS
5) Sloj pristupa/prezentacije - zadatak mu je objediniti postojeće servise na razini postojećeg sučelja
6) Sloj integracije - integrirati sve prethodne slojeve sa bazom podataka, aplikacijama itd.
7) Sloj osiguranja kvalitete usluga, sigurnost, upravljanje, nadzor - integracijski i upravljački sloj koji upravlja cjelokupnom arhitekturom

-SOA – arhitektura web servisa



Standardi i protokoli – sloj zadužen za funkcioniranje i rad servisa
Servisna infrastruktura – sloj koji predstavlja zajedničke sustave koji su potrebni za upravljanje i funkcioniranje servisa, upravlja transportom podataka odnosno XML poruka
Aplikacijski servisi – sloj u kojem je smještena programska logika pojedinog web servisa

SOAP – Simple Object Access Protocol
-komunikacijski protokol koji služi za komunikaciju web servisa



HTTP protokol nosi SOAP protokol, TCP/IP protokol nosi HTTP protokol

-Hub and spoke

-integracijski mehanizam, sastoji se od središnjeg čvora – hub-a, na njega su vezane sve komponente SOA (aplikacije, baze, repozitoriji, upravljački mehanizmi itd.)

-sva komunikacija ide preko središnjeg čvora (provjera prava pristupa, podataka, rezultata)

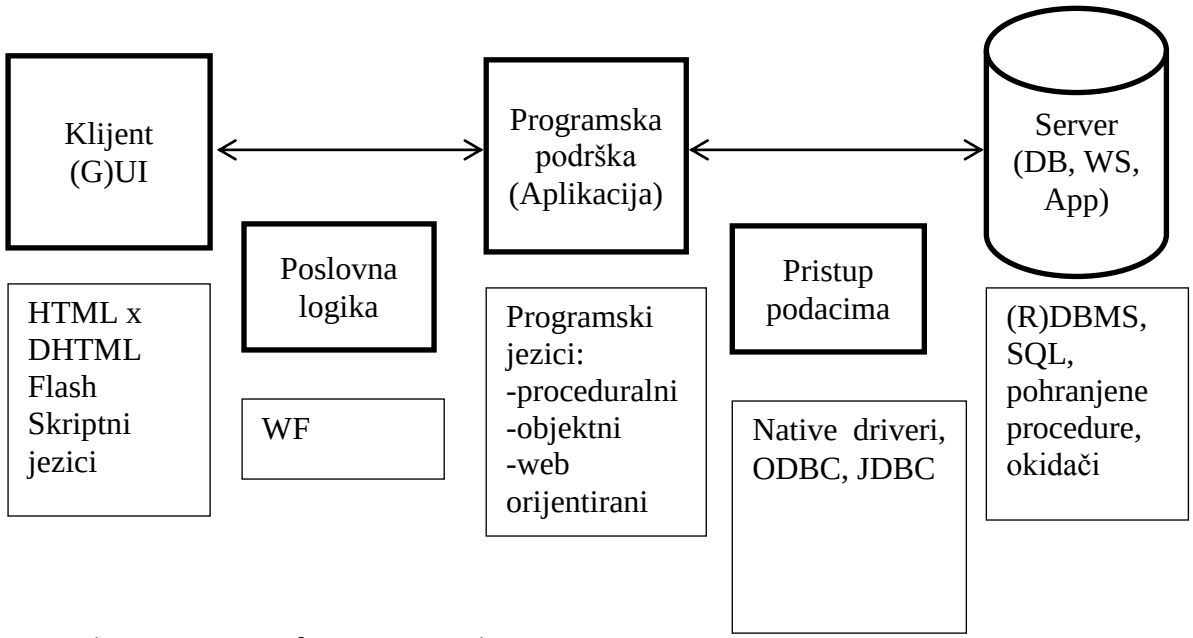
Enterprise Servise Bus

-rad na principu sabirnice

-komponente povezane međusobno pomoću „adaptera“

-omogućuje veliki propusnost

Osnovni elementi arhitekture IS-a



HTML (HyperText Markup Language)

-prezentacijski jezik za izradu web stranica

-trenutna najnovija inačica – HTML 5

DHTML (Dynamic HTML)

-jezik za izradu web stranica koristeći više jezika

-ugrađen u 4. generaciju pretraživača

Flash

-služi za izradu grafičke animacije

-izrađuje se u slojevima (sloj zvuka, sloj slike, sloj teksta itd.)

Skriptni jezici

-npr. Javascript

- programski sloj prebačen sa strane servera na stranu klijenta zbog ubrzanja rada

-npr. ispravnost e-mail adrese – kod raslojava i provjerava svaki znak u e-mail adresi

WF
-dijagram radnih tokova
-notacija za podešavanje slijeda odvijanja procesa

Proceduralni jezik
-npr. Turbo Pascal
-jezik sa kojim „govorimo“ računalu kako će nešto biti izvršeno

Objektni jezik
-npr. Python, C++, C#, Java
-programski jezik temeljen na objektima

ODBC
objašnjeno na 2. stranici

Okidači
-npr. automat koji zove kada nismo platili račun
-SQL upit na bazu podataka
-aktivira se po nastupu unaprijed određenog događaja

2. RAZMJENA PODATAKA U E-POSLOVANJU

-Ciljevi razmjene podataka elektroničkim putem:

- Smanjenje troškova
- Povećanje produktivnosti
- Real time razmjena informacija

-Temeljni cilj je zadržavanje integriteta i sigurnosti podataka uz prijenos u standardiziranom i razumljivom formatu
-načini realiziranja:

- Formatirane datoteke - datoteke precizno utvrđene strukture, definiran naziv, vrsta i veličina polja)
- EDI (Electronic Data Interchange) – razmjena poslovnih dokumenata (primjena u B2B segmentu), skup je, nekompatibilan, kompleksan te ima probleme integracije
- XML

XML (Extensible Markup Language)
-opisni jezik – označava značaj pojedinog podatka koji se razmjenjuje
-„Extensible“ – tagovi nisu unaprijed definirani, tagove definira krajnji korisnik
-rješenje za EDI koji će biti jednostavniji, besplatan, rasprostranjeniji
-omogućuje razmjenu i pohranu podataka
-Cilj – održavanje fleksibilnosti formatiranih datoteka kroz elegantnije rješenje
-rad s XML-om:

- Definiranje strukture elemenata (tagova)
  - DTD – starija varijanta, jednostavniji
  - XML Schema - složeniji
- Zapis sadržaja primjenom elemenata (tagova)

-XML dokument mora biti:

- Dobro formatiran (sukladan pravilima sintakse)
- Valjan (odgovara pravilima definiranim od strane korisnika ili XML Scheme)

1) Definiranje strukture (DTD)

```
<!ELEMENT osobna_iskaznica(osoba*)>
<!ELEMENT osoba (br_osobne, ime_prezime, spol, drzavljanstvo, dat_rodenja, vrijedi_do)>
<!ELEMENT br_osobne (#PCDATA)>
<!ELEMENT ime_prezime (#PCDATA)>
<!ELEMENT spol (#PCDATA)>
<!ELEMENT drzavljanstvo (#PCDATA)>
<!ELEMENT dat_rodenja (#PCDATA)>
<!ELEMENT vrijedo_do (#PCDATA)>
```

1) Definiranje strukture (XML Schema)

```
<xs:schema xmnls:xs=http://xxx>
<xs:element name:“osobna_iskaznica“ type=“Osobna_iskaznica“/>
<xs:complexType name=“Osobna_iskaznica“/>
<xs:sequence>
        <xs:element name=“osoba“ type=“Osoba“/>
        <xs: complexType name=“Osoba“>
        <xs:sequence>
                <xs:element name=“br_osobne“ type=“xs:string“/>
                <xs:element name=“ime_prezime“ type=“xs:string“/>
                <xs:element name=“spol“ type=“xs:string“/>
                <xs:element name=“drzavljanstvo“ type=“xs:string“/>
                <xs:element name=“dat_rodenja“ type=“xs:string“/>
                <xs:element name=“vrijedi_do“ type=“xs:string“/>
        </xs:sequence>
        </xs:complexType>
</xs:sequence>
</xs:complexType>
</xs:schema>
```

2) Zapis sadržaja

```
        <osobna_iskaznica>
                <osoba>
                        <br_osobne>193423324<br_osobne>
                        <ime_prezime>Cvjetko Livadić<ime_prezime>
                        <spol> M<spol>
                        <drzavljanstvo>Hrvatsko<drzavljanstvo>
                        <dat_rodenja>07.07.1977.<dat_rodenja>
                        <vrijedi_do>07.07.2015.<vrijedi_do>
                </osoba>
        </osobna_iskaznica>
```

XML specifikacije
-unaprijed definirane XML scheme po pojedinim područjima djelatnosti (standardizacija)

ebXML (e-Business using XML)
-cilj je osigurati interoperabilan, siguran i nepromjenjiv način korištenja poslovnih informacija

cXML (Commerce XML)
-razmjena podataka u e-businessu (najrašireniji B2B protokol)
-pruža posebne sheme za standardne poslovne transakcije

finXML
-jedinstven standard za razmjenu podataka na tržištu kapitala
-omogućuje razmjenu kompliciranih i strukturiranih financijskih dokumenata i transakcija

XSL (Extensible Stylesheet Language)
-sastoji se od 3 dijela koji pružaju dodatnu funkcionalnost XML-a:

- XSLT – jezik za transformaciju XML dokumenta (svođenje različitih schema na isto)
- XPath – jezik na navigaciju unutar XML dokumenta („pretraživanje“ i navigacija kroz elemente i atribute XML-a)
- XSL-FO – jezik za formatiranje XML dokumenta (prikaz i formatiranje prikaza XML podataka za izlaz)

UBL (Universal Business Language)
-nastaje kao inicijativa standardizacije XML scheme i najčešćih poslovnih procesa

3. ELEKTRONIČKI (DIGITALNI) POTPIS

-digitalni kod koji služi za zaštitu poruke, dokumenata odnosno omogućuje elektroničku identifikaciju i autentičnost sadržaja poruke
-mehanizam koji pomaže u očuvanju integriteta i autentičnosti razmijenjenih sadržaja
-napredni el-potpis ima istu pravnu snagu kao i vlastoručni potpis i otisak pečata (ako je izrađen u skladu sa zakonom)
-definicija ZEP-a: elektronički potpis je skup podataka u elektroničkom obliku koji su pridruženi ili su logički povezani s drugim podacima u elektroničkom obliku i koji služe za identifikaciju potpisnika i vjerodostojnosti potpisanog elektroničkog dokumenta
-u RH upotreba e-potpisa je regulirana Zakonom o elektroničkom potpisu

-nastaje kao rješenje za probleme pisanog potpisa:

- Problem krivotvorenja
- Problem autentičnosti
- Problem ponovnog korištenja
- Problem promjene izvornog dokumenta
- Problem poricanja

-elektronički potpis karakterizira:

- Ne može se krivotvoriti (radi na mehanizmu privatnog i javnog ključa, samo pošiljatelj zna svoj privatni ključ)
- Autentičan je (privatni ključ je svojstven jednoj osobi)
- Nije ga moguće ponovno koristiti (potpis jednog dokumenta vrijedi samo za taj dokument zbog načina kreiranja)
- Potpisani dokument je nepromjenjiv (promjena samo jednog bita i potpis više neće vrijediti)
- Ne može se poricati

-temelji se na:

- HASH algoritmu (provjera integriteta, algoritam za kreiranje sažetka, programski algoritam)
- Algoritam za šifriranje (osiguranje provjere pošiljatelja/autora)

-privatni ključ – služi za šifriranje i dešifriranje

-javni ključ – tajni i javni ključ, sadržaj šifriran tajnim ključem može biti dešifriran isključivo javnim ključem

Princip rada digitalnog potpisa

-princip rada digitalnog potpisa se odvija u 5 koraka:

1. Na datoteku koji želimo digitalno potpisani primjenjuje se hash algoritam. Hash algoritam na temelju datoteke kreira njezin sažetak (digitalni kod točno određene veličine koji je nastao primjenom određenog algoritma). Sažetak je važan za očuvanje integriteta poruke (datoteke).
2. Sažetak potpisujemo sa svojim privatnim ključem – nastaje digitalni potpis (tzv.šifrirani sažetak datoteke)
3. Šaljemo originalnu poruku, digitalni potpis iz 2. koraka te certifikat (digitalna isprava koja jednoznačno identificira pojedinca, računalo, mrežnu adresu itd. koja posjeduje javni ključ – potreban je jer sadrži podatke o potpisniku, javnom ključu, koji su se hash algoritmi i algoritmi za kriptiranje koristili)
4. Primatelj na primljenu poruku primjenjuje hash algoritam i kreira sažetak.
5. Primatelj uzima digitalni potpis i iz certifikata čita algoritam šifriranja i javni ključ, te dešifrira digitalni potpis – dobiva sažetak iz prvog koraka. Primatelj zatim uspoređuje sažetke iz koraka 4 i iz koraka 5 te ako su isti onda se radi o autentičnoj poruci. Ako nisu isti onda je došlo do pogreške u prijenosu poruke, netko je namjerno promijenio poruku ili to nije poruka od originalnog pošiljatelja

HASH algoritam

-algoritam sažimanja kojim se varijabilni ulaz proizvoljne duljine transformira u niz znakova fiksno određene duljine

-karakteristike:

1. Niz ulaznih podataka je proizvoljne dužine
2. Izlazni podatak je stalne veličine
3. Nemoguće je izvesti inverznu funkciju (ne može se na temelju sažetka doći do originalne datoteke)
4. Ne daje dva ista izlaza za dva različita ulaza (sažetak različitih datoteka će uvijek biti različit)

-Kolizija – svojstvo da za 2 različita ulaza dobijemo isti sažetak

Princip rada hash funkcije

-princip rada hash funkcije se odvija u nekoliko koraka:

1. Funkcija dijeli poruku na blokove veličine 512 bita
2. Svaki blok od 512 bita se dijeli na 4 bloka od 128 bita
3. Svaki blok od 128 bita se dijeli na 4 bloka od 32 bita
4. Provodi se 16 sličnih operacija baziranih na nelinearnoj funkciji, modularnom zbrajanju i rotaciji bitova ulijevo

SHA algoritam (Secure Hash Algoritam)
-algoritam koji je dio SHS (Secure Hash Standard) standarda
-na osnovu maksimalne duljine poruke od 264 bita izračunava broj duljine 164 bita

RSA algoritam (Rivest, Shamir, Adleman)
-algoritam koji spada u sustave s javnim ključem

Djelovanje kriptosustava s javnim ključem
Nemam blage kako ni zašto, al sam stavio za svaki slučaj

- Kriptiranje se vrši tajnim ključem (razgovjetni tekst se kodira u niz cijelih brojeva *Mi* koji smiju poprimiti vrijednosti iz intervala od 0 do N-1. Svaki od tih cijelih brojeva se kriptira u $Ci=(Mi)emodN$
- Dekriptiranje se obavlja tako da se *Ci* dekriptira u razgovjetni oblik $Mi = (Ci)dmodN$ te se niz brojeva *Mi* dekodira u izvorni tekst

Princip rada RSA algoritma

-Princip rada se odvija u 5 koraka, sličan princip kao i kod digitalnog potpisa:

1. Pošiljatelj računa sažetak pomoću hash funkcije
2. Pošiljatelj kreira sažetak pomoću tajnog ključa, te tako kreira digitalni potpis
3. Pošiljatelj šalje originalni dokument i digitalni potpis
4. Primatelj dobiva potpisanu poruku, a iz originalne poruke izračunava sažetak
5. Primatelj dekriptira digitalni potpis pošiljateljevim javnim ključem te uspoređuje dekriptirani sažetak sa onim koji je sam izračunao. Ako su jednaki, poruka je autentična i njezin integritet je očuvan.

XML digitalni potpis
-omogućuje potpisivanje samo određenog dijela dokumenta, a ne samo cijelog dokumenta kao kod običnog i naprednog elektroničkog potpisa.
-vrste XML digitalnog potpisa:

1. Detached – potpisani podaci i XML potpis su odvojeni
2. Enveloping – potpisani podaci su ugrađeni u XML potpis
3. Enveloped – potpis je ugrađen u podatke koje potpisuje

-osnovni element XML digitalnog potpisa je Signature element <Signature ID?>

-Struktura/shema XML digitalnog potpisa:

<Signature ID?> - korijenski element definicije digitalnog potpisa
<SignedInfo> - služi za definiranje primjenjenog kanonizacijskog algoritma i algoritam korišten kod potpisivanja
<CanonicalizationMethod/> - zapis naziva korištenih algoritama
<SignatureMethod/> - zapis naziva korištenih algoritama
(<Reference URI? > - sadrži putanju/adresu resursa na koje se primjenjuje digitalni potpis

(<Transforms>)? – definira sadržaj koji je korišten za provođenje niza transformacija, pretvorbe sadržaja
<DigestMethod> - definira naziv algoritma koji se koristi
<DigestValue> - služi za pohranu kreiranog sažetka u okviru XML digitalnog potpisa
</Reference>)+ - zatvaranje elementa reference
</SignedInfo> - zatvaranje elementa signedinfo
<SignatureValue> - zapis stvarne vrijednosti digitalnog potpisa
(<KeyInfo>)? – poveznice na primijenjene ključeve, certifikate i sl.
(<Object ID?>)*
</Signature> - zatvaranje elementa signature

-Kanonizacija:
Pročišćivanje XML sadržaja (čišćenje kvačica i sl. – č, ć, š itd.)
Ujednačavanje XML schema

Kreiranje XML digitalnog potpisa je isto kao i kod digitalnog potpisa, zato neću opet pisat

MIKROPLAĆANJA

-sustavi plaćanja namijenjeni za plaćanje niskih iznosa (nekoliko centi)
-zamjena za gotovinu:

1. Jeftiniji
2. Brži
3. jednostavniji za izračun, provjeru i sljedivost

-omogućuju tehnološke uštede – sustavi sa nižom razinom sigurnosti i enkripciju:

1. ne radi se provjera transakcija
2. simetrična enkripcija (šifriranje i dešifriranje pomoću jednog ključa)
3. pretplaćanje
4. grupiranje (agregacija narudžbi tj. transakcija koje se šalju dnevno, tjedno te parcijalna anonimnost)

-Skupine sustava:
<u>1) Pretplaćene kartice (Prepaid cards)</u>
- ne izdaju ih banke
- predstavljaju uslugu u budućnosti – na karticama imamo sredstva koje ćemo trošiti u budućnosti
- nisu novac jer su iskoristivi samo kod jednog prodavatelja

<u>2) Elektronički novčanik</u>
- izdaje ih banka
-prezentira stvarni novac
- oblici:
kartica (za licem u lice ili internet)
virtualni oblik

<u>Pametne Kartice</u>
- sadrže čip koji je tzv. mikroračunalo
-svaki dio čipa obavlja svoju funkciju:

1. napajanje (4-5V)
2. uzemljenje
3. reset
4. sat
5. program
6. sučelje za ulaz/izlaz

-ispod toga je mikroprocesor sa RAM i ROM memorijom koji sadrže podatke o kartici, pinu, stanju itd.

GeldKarte

-smartcard sustav

-kartica sadrži brojače koji prezentiraju novčanu vrijednost

-kartica se puni preko loading terminala (tereti se korisnikov bankovni račun)

-trošenje se odvija na terminalima ili na internetu

-za zaštitu koristi 126-bitni hash algoritam, svaka kartica ima jedinstveni ID, ključ, PIN i broj transakcije

-Postupak plaćanja

*(*nije spominjao na predavanjima, ali da otprilike znamo kako funkcionira sustav. Izbacite si ovo ako hoćete*)*

1. kupac ubacuje karticu u utor
2. trgovac autorizira karticu
3. transfer vrijednosti kupnje
4. generira se el.račun
5. trgovac prezentira račun banci da dobije sredstvo na vlastiti račun

-Postupak autorizacije:

*(*nije spominjao na predavanjima, ali da otprilike znamo kako funkcionira sustav. Izbacite si ovo ako hoćete*)*

1. trgovčev SAM generira slučajni RAND broj (sprječavanje replay attacka) i šalje ga na kupčevu karticu sa zahtjevom ID kartice
2. kartica šalje ID, generirani sekvencijalni broj, RAND i H(CID) kriptiran simetričnim tajnim ključem SKc
3. Nema enkripcije javnim ključem
4. Trgovac izračunava SKc iz CID-a i vlastitog tajnog ključa SKm
5. Trgovac sada može provjeriti integritet poruke kartice izračunavanjem H(CID)

Quick sustav

-smartcard sustav

-Postupak kupnje:

*(*nije spominjao na predavanjima, ali da otprilike znamo kako funkcionira sustav. Izbacite si ovo ako hoćete*)*

1. Trgovčev server kontaktira server plaćanja, odašilje klijentovu IP adresu i vrijednost transakcije, kratak opis robe i ID trgovca
2. Server plaćanja zaključava trgovca za transakciju, kontaktira „novčanik“ preko TCP-a na posebnom portu napravljenom za Quick Internet plaćanje. Klijent računalo pristupa čitaču i traži kupčevu karticu
3. Prije nego što se kartica tereti sa iznosom, klijent prikazuje poruku koja opisuje naručena dobra i ukupan iznos i dozvoljava kupcu da odustane

RFID (Radio-frequency identification)

- tehnologija korištena u karticama (npr. skladišta, studentski dom, FOI)

-tanka nit koja na sebi ima zapis i čitač čita taj zapis

-temelji se na komuniciranju dvaju uređaja na udaljenosti

Upotreba mobilnih uređaja

1. Uređaj kao čitač – square čitač koji emulira POS terminal
2. SMS kao sredstvo plaćanja – mVIP, mPAY, usluga reg.kod telekom operatera i banke
3. Beskontaktan (PayPass, Google Wallet)

PayPass

- temelji se na RFID tehnologiji zbog svoje prilagodljivosti
- automatizirana autorizacija (prislanjanje na čitač, nema PIN-a)
- sigurnosni problem (nema provjere vlasnika ili transakcije)

Google Wallet

- umjesto kartice – smartphone uređaj sa aplikacijom GW
- autorizacija svake transakcije PIN-om
- postoje 2 oblika – aplikacija na tabletu/smartphoneu, internet plaćanje uz google servise
- offline i online način rada

Sustavi mikroplaćanja:

1. PayWord
2. MicroMint
3. Milicent

Uloge kod mikroplaćanja

Posrednik – pr. Paypal; sustav koji izdaje elektronički novac
Dobavljači – prodavač; od njega kupujemo
Korisnik – mi; krajnji kupac

PayWord

- sustav mikroplaćanja kod kojeg koristimo „payword“ za plaćanje
-temelji se na plaćanju informacijama - stringovima koji imaju apoensku vrijednost.
- Korištenje Payword-a (proces kupnje):
    1. Korisnik kontaktira posrednika preko sigurnog kanala (npr. SSL) te uspostavlja Payword račun kod posrednika (plaća sa stvarnim novcem)
    2. Posrednik izdaje korisniku virtualnu (pretplatničku) karticu (certifikat; sadrži ime posrednika, korisnika, korisnička IP adresa, korisnikov javni ključ)
    3. Certifikat ovlašćuje kupca kod dobavljača
    4. Korisnik kreira payword lance (tipične duljine 100 jedinica) za određenog dobavljača

-Svojstva:

1. Plaćanje i verifikacija kupca od strane dobavljača provodi se offline (nema treće strane)
2. Niz za plaćanje ne otkriva pojedinačne vrijednosti
3. Prijevaru od strane korisnika otkriti će posrednik, gubitak će biti malen
4. Dobavljač pohranjuje zapis važećih payword-a kako bi se zaštitio od ponovljenog odašiljanja

MicroMint

-sustav mikroplaćanja koji ima namjeru napraviti sredstvo plaćanja (kovanice) koje su jednostavne za provjeru a teške za kreiranje
-posrednici proizvode „kovanice“ kratkog vijeka trajanja i prodaju ih korisnicima
-korisnici plaćaju dobavljačima putem „kovanica“
-dobavljači zamjenjuju kovanice sa posrednicima za pravi novac

MicroMint Sustav:

1. Kupac naručuje nove kovanice i vraća nepotrošene posredniku
2. Kupac troši kovanice, plaća dobavljaču
3. Posrednik izdaje nove kovanice kupcu
4. Dobavljač zamjenjuje kovanice za druge vrijednosti od posrednika
5. Dobavljač prenosi informacije kupcu

Milicent

- sustav mikroplaćanja
-dobavljač proizvodi vlastite, specifične zapise i prodaje ih posrednicima za stvarni novac uz popust
-posrednici prodaju zapise različitih dobavljača različitim korisnicima
-zapis je pretplaćen – obećanje usluge dobavljača u budućnosti

Milicent sustav:

1. Korisnik kupuje posrednikove i troši dobivajući dobavljačeve zapise od posrednika
2. Posrednik izdaje posrednički zapis korisniku, zamjenjuje dobavljačev zapis za posrednički, surađuje s bankama, prikuplja sredstva od korisnika, plaća dobavljačima
3. Dobavljač prodaje zapise posrednicima te prihvaća dobavljačeve zapise od korisnika

Problem plaćanja enkripcijskim stringovima

1. Ponavljano trošenje već iskorištenih stringova
2. Slijepi potpis koji sakriva sadržaj i osigurava anonimnost

Slijepi potpis

-predstavljaju temelj anonimnosti u elektroničkom novcu
-potvrđuje autentičnost povjerljivih podataka i štiti anonimnost
-Proces:

1. Osoba A želi da osoba B potpiše poruku M
2. A množi M s brojem (maskirajući faktor)
3. A šalje maskiranu poruku B
4. B potpisuje maskiranu poruku sa svojim privatnim ključem, šalje natrag A
5. A dijeli sa maskirajućim faktorom, sada ima M potpisan od B

Chaum Double-Spending Protocol

-protokol za očuvanje sigurnosti e-plaćanja, odnosno sprječavanje ponovljenog trošenja novca

-Računanje:

1) Broj računa pretvorimo u heksadekadski oblik (npr. broj računa 12 = hex 0C)

$12_{(10)} = 00001100_{(2)}$

$00001100_{(2)} = 0C_{(16)}$

2) Uzmemo serijski (100) i maskirajući (5) broj

3) Od banke tražimo kovanicu sa serijskim brojem 100 x 5 = 500

4) Odabiremo slučajan broj b i kreiramo b slučajnih brojeva (b = 6)

5) Radimo XOR svakog slučajnog broja sa vlastitim brojem računa

i	Račun	Slučajni broj	Račun XOR slučajni broj
0	0C	1B	17
1	0C	13	1F
2	0C	09	05
3	0C	05	09
4	0C	2B	27
5	0C	11	1D

6) Osoba B zaprima kovanice od nas. Pronalazi b te odabire slučajan broj sa b bitova (npr.111010)

7) Za svaku poziciju bita u kojoj B broj ima 1, on zaprima slučajan broj od A za tu poziciju

8) Za svaku poziciju bita u kojoj B broj ima 0, on zaprima račun XOR slučajni broj A za tu poziciju.

i	Račun	Slučajni broj	Račun XOR slučajni broj	B bit	B zaprima
0	0C	1B	17	0	17
1	0C	13	1F	1	13
2	0C	09	05	0	05
3	0C	05	09	1	05
4	0C	2B	27	1	2B
5	0C	11	1D	1	11

9) Osoba B šalje zadnji stupac u banku kada polaže kovanicu

10) Mi pokušamo potrošiti kovanicu ponovno kod osobe C. On pronalazi b=6 i odabire slučajni broj 010000

11) Osoba C prolazi istu proceduru kao B i šalje brojeve koje je primio u banku kamo polaže kovanicu

i	Račun	Slučajni broj	Račun XOR slučajni broj	C bit	C zaprima

0	0C	1B	17	0	17
1	0C	13	1F	0	1F
2	0C	09	05	0	05
3	0C	05	09	0	09
4	0C	2B	27	1	2B
5	0C	11	1D	0	1D

12) Banka odbija platiti osobi C jer je kovanica već potrošena kod osobe B.
13) Banka kombinira podatke od osobe B i osobe C i korištenjem XOR gdje je pronašla podatke iz dva izvore identificira nas kao varalicu

i	Račun	Slučajni broj	Račun XOR slučajni broj	Slučajni XOR račun XOR slučajni	Vlasnik računa
0	0C		17		
1	0C	13	1F	0C	A
2	0C		05		
3	0C	05	09	0C	A
4	0C	2B			
5	0C	11	1D	0C	A

POJMOVI (spomenuti na predavanju)

>Strojni jezik – naredbe računalu na razini aritmetičko-logičke jedinice

>Datotečni sustav – određuje kako će se podaci zapisivati (FAT32, NTSF, Joliet)
>CRUD – create, read, update, delete – prava pristupa određenoj tablici koju korisnik ima ili dodjeljuje drugim korisnicima
>Čvrsta povezanost - komponente međusobno ovise jedna o drugoj (npr. bankomat – ne može se isplatiti novci ukoliko autorizacija nije provedena)
>Kapsulacija – programska ovojnica oko komponente, komponenti pristupamo spajanjem na njezinu ovojnicu
>SSO (Single Sign On) – prijavljivanje na sve komponente samo jednom (prijavljivanje u google automatski prijavljuje korisnika u sve ostale komponente – youtube, gmail, google drive itd.)
>Orkestracija – zadatak joj je povezivanje svih web servisa prema definiranom tijeku u smislenu cjelinu
>Koreografija - zadatak joj je povezivanje svih web servisa prema definiranom tijeku u smislenu cjelinu
>Sinkroni – sinkronizacija web servisa, radi u realnom vremenu (online)
>Akroni – sinkronizacija web servisa, radi u offline načinu, sinkronizacija se provodi nakon uspostavljanja mreže
>Privatni ključ – služi za šifriranje i dešifriranje
>Javni ključ – tajni i javni ključ, sadržaj šifriran tajnim ključem može biti dešifriran isključivo javnim ključem