SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1748

PROGRAMSKA PODRŠKA ZA DVODIMENZIJSKU GRAFIKU NA WEB-U

Merlin Rebrović

Zagreb, rujan 2008.

Posvećujem ovaj rad baki Dragici.

Sadržaj

1.Uvod.....1

2.Vektorski XML zapis.....2

2.1.Razvoj tekstualnih vektorskih formata i nastanak SVG-a.....2

2.2.SVG dokument.....3

2.3.Animiranje SVG-a.....9

3.Web aplikacije.....11

4.Programsko ostvarenje.....14

4.1.Ideja i tehnologija.....14

4.2.Struktura i način rada aplikacije.....14

4.3.Ulazni podaci.....17

4.4.Izlazni podaci.....20

4.5.Linijski graf.....23

4.6.Stupčasti graf.....25

4.7.Histogram.....27

4.8.Kružni graf.....28

4.9.Dodatne mogućnosti.....30

5.Zaključak.....33

6.Sažetak / Abstract.....34

7.Popis oznaka i kratica.....35

8.Popis tablica.....36

9.Popis slika.....37

10.Literatura.....38

1. Uvod

Pojavom prvih grafičkih web preglednika web stranice su počele dodavati grafiku različitih formata. Ta je grafika većinom bila rasterska, a vektorska se pojavljivala uglavnom kao popratni sadržaj – tablice, granice između odjeljaka, podvučeni ili prekriženi tekst i slično. Vektorska je grafika bila jako zapostavljena i prečesto zamjenjivana rasterskom, čak i na mjestima gdje je to najmanje odgovaralo.

U posljednjem je desetljeću od strane velikih kompanija dogovoren novi vektorski tekstualno baziran format koji je prvenstveno namijenjen web upotrebi - Scalable Vector Graphics (SVG). Pojavom ovog formata stvorena je mogućnost jednostavnog i kvalitetnog kreiranja vektorskih grafika koje su skalabilne, neovisne o platformi i otvorene svakome tko ih želi istražiti.

U isto vrijeme se na Internetu javlja trend pojave velikog broja servisa koji su često jednostavnije verzije komercijalnih alata, npr. tekst procesori, klijenti elektroničke pošte, Internet bankarstvo i sl. Web servisi vezani uz grafiku su još uvijek rijetki i vrlo često vezani uz rastersku grafiku.

Ovaj rad će pokušati spojiti SVG format i trend web servisa u web aplikaciju koja će služiti za iscrtavanje jednostavih grafova.

2. Vektorski XML zapis

2.1. Razvoj tekstualnih vektorskih formata i nastanak SVG-a

Na samom kraju 20. stoljeća World Wide Web Consortium (W3C) je osnovao SVG radnu grupu kako bi pronašao alternativu PostScript formatu zapise koji je mogao koristiti skalabilne fontove i objekte, no s velikim utjecajem na veličinu dokumenta. Sredinom 1998. godine Adobe, IBM, Netscape i Sun Microsystems su predložili PGML (Precision Graphics Markup Language), a Hewlett Pacard, Macromedia, Microsoft i Visio su predložili VML (Vector Markup Language). SVG radna grupa je u nekoliko mjeseci napravila prvi predložak SVG formata uzimajući u obzir oba predloška. Odlučeno je da će novi format koristiti XML kao zapis i da će biti optimiziran za Web.

Te su odluke donesene zbog dva razloga. Prvi je dolazak XML standarda u 1998. godini. U tom se trenutku otvorila mogućnost napraviti tekstualno baziran vektorski grafički zapis u već postojećim i standardiziranim tehnologijama. Tekstualni format donosi neke radikalne promjene. Prvo, čitljiv je ljudima bez potrebe za posebnim alatima, no opet je prilagođen analizi i parsiranju računalom. Drugo, izvorni kod/zapis je uvijek moguće pogledati. SVG je od početka izgrađen u duhu otvorenog koda (open source). Mnogi su se bunili zbog nemogućnosti zaštite autorskih prava u novom formatu jer svatko može kopirati cijeli ili samo potrebni dio rada. Isti ljudi su bili brzi na zaključku da SVG neće funkcionirati, no ista stvar se dogodila dolaskom HTML (HyperText Markup Language). Za svaku stranicu na Internetu je moguće pogledati njen izvorni kod. To je donijelo novu perspektivu na razvoj tehnologije i inspekciju sadržaja i ubrzalo je širenje i edukaciju korisnika i programera.

Velika većina postojećih vektorskih formata je binarnog tipa, zatvorena i zaštićena od proizvođača programskog paketa za obradu grafike. Takve je zapise uglavnom moguće koristiti samo uz jednu, najčešće komercijalnu,

aplikaciju i prebacivanje u drugi format može biti teško, nekvalitetno, a ponekad i nemoguće. Jedan od poznatijih formata sličan SVG-u koji se vrlo često koristi na Webu je Adobe Flash. Na Flash se odnose sve gore navedene mane zatvorenih i binarnih formata. Uz sve te mane Flash ipak nudi najveću mogućnost prikaza sadržaja na Internetu, no zbog svoje prirode nikad nije preuzeo veći dio tržišta. Svi web preglednici nude ugradnju dodatnih programa koji omogućava pregledavanje Flasha, no nitko ne želi nativno ugraditi podršku. SVG ima potencijala to promijeniti. SVG je otvoreni standard na koji se svi mogu osloniti. SVG podržava skoro sve mogućnosti Flasha, a nisu potrebni nikakvi posebni, komercijalni i skupi alati za njegovu obradu. Zbog svih ovih mogućnosti skoro svi moderni web preglednici su već implementirali većinu SVG specifikacije.

2.2. SVG dokument

Svaki SVG dokument je istovremeno i XML dokument. Zato mora poštovati neke XML standarde kao što su strukturiranost (well-formedness) i valjanost (validity) i uz to zadana pravila propisana SVG standardom. XML je kroz godine korištenja pokazao svoje vrline i mane.

Neke od vrlina:

- Čitljiv je ljudima.
- Podržava Unicode čime se može zapisati skoro svaka informacija u svim ljudskim jezicima.
- Može prikazati učestale računalne strukture: zapise, liste i stabla.
- Format opisuje samog sebe.
- Ima strogu sintaksu čime je olakšano analiziranje podataka.
- Baziran je na međunarodnim standardima.
- Neovisan je o platformi.

Neke od mana:

- XML sintaksa je redundantna ili vrlo velika u usporedbi s binarnim formatima koji prikazuju slične podatke.
- Postoje manje redundantni formati koji su čitljivi ljudima.
- Hijerarhijska struktura nije pogodna za sve strukture podataka.
- Razlika između sadržaja elemenata i atributa nekad nije očita što otežava dizajn podatkovnih struktura.

```
<?xml version="1.0" standalone="yes"?>
<!-- ovo je komentar -->
<studentskiRadovi>
     <rad tip="seminar">
         <student>Merlin Rebrović</student>
         <mentor>Željka Mihajlović</mentor>
         <naziv>SVG na Webu</naziv>
     </rad>
     <rad tip="diplomski">
         <student>Merlin Rebrović</student>
         <mentor>Željka Mihajlović</mentor>
         <naziv>Programska podrška za dvodimenzijsku
     grafiku na Web-u</naziv>
     </rad>
</studentskiRadovi>
```

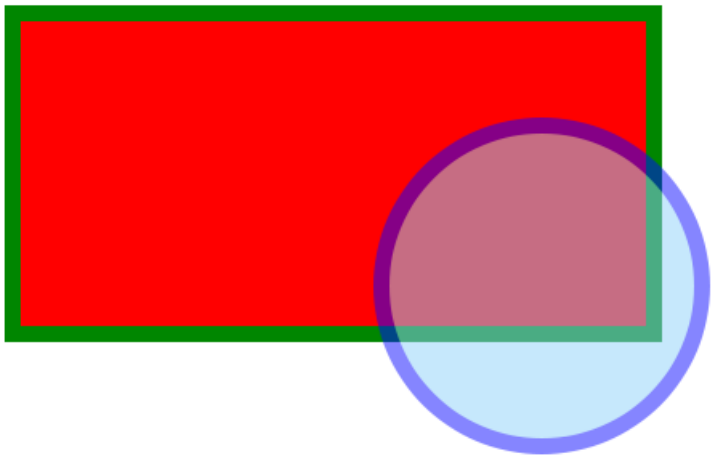
Primjer 1: Jednostavni XML dokument.

Svi redovi koji započinju s `<?xml` su XML direktive. One su opcionalne, no pomažu prilikom pregleda dokumenta. Redak `<?xml version="1.0" standalone="yes"?>` nam govori da se radi o XML-u verziji 1.0 i da nema dodatnih dokumenata bez kojih bi prikaz ovog dokumenta bio nemoguć.

Još jedna opcionalna direktiva je `<!DOCTYPE ...` koja definira lokaciju DTD dokumenta kojim se može provjeriti valjanost SVG dokumenta. Kao što se vidi iz istog retka, DTD sadrži sintaksu SVG specifikacije verzije 1.1.

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink">

    <rect x="30" y="30" width="400" height="200"
        fill="red" stroke="green" stroke-width="10" />
    <circle cx="360" cy="200" r="100" opacity="0.5"
        fill="lightskyblue" stroke="blue" stroke-
width="10" />
</svg>
```

Primjer 2: Jednostavni SVG dokument. Po strukturi se može vidjeti da je SVG zapravo XML jezik. (BasicSVGDocument.svg)



Slika 1: Prikaz primjera 2 u web pregledniku s podrškom za SVG. Treba primjetiti redoslijed kojim su deklarirani elementi i redoslijed iscrtavanja istih.

Informacije o samim oblicima započinju prvom pojavom `<svg>` elementa. `<svg>` element u sebi među ostalim atributima sadrži i definicije o "dosegu imena" (eng. namespace). Redak `<svg xmlns="http://www.w3.org/2000/svg"></svg>` je nužan dio svakog SVG dokumenta, a ujedno sam za sebe tvori i najmanji. Prvi `<svg>` element je korijenski element XML dokumenta i unutar njega se nalaze sve ostale informacije.

SVG koristi "slikarski model" iscrtavanja. Boja se postupno primjenjuje na izlazni uređaj tako da svako nanošenje boje prekrije dio tog uređaja. Kada se prostor preklapa s prethodno naslikanim prostorom, novo naslikano područje djelomično ili potpuno prekriva staro. U slučaju da je novo područje djelomično ili potpuno prozirno, računa se i prikazuje kompozicija boja starog i novog područja.

Elementi u SVG dokumentu imaju implicitni red iscrtavanja i to tako da se prvi elementi u dokumentu prvi iscrtavaju. Sljedeći elementi se iscrtavaju preko starih elemenata. Grupe u raznim spremnicima (npr. spremnik `<g>`) se prvo iscrtaju na privremeno odvojeno područje, a zatim se cijela grupa kao jedan element iscrta na izlazni uređaj.

SVG podržava sljedeći skup osnovnih oblika: pravokutnik, krug, elipsa, linija (niz linija) i poligon. Svi osnovni oblici se mogu dobiti korištenjem krivulja, no zbog jednostavnosti i intuitivnosti atributa tih oblika su osnovni oblici često korišteni u jednostavnijim grafikama. Krivulje su vrlo vjerojatno najvažniji oblik u SVG formatu, a svakako najsvestranije. One rade na principu trenutne točke. Početna točka krivulje je na poziciji (0,0) i do svake sljedeće zadane točke se povlači i iscrtava dio krivulje. Taj dio ovisi o zadanim parametrima krivulje i njenom putu.

Sintaksa koja opisuje krivulju je vrlo sažeta kako bi veličina dokumenta bila što manja. U kompleksnijim SVG dokumentima većinu prostora ispunjava upravo "d" atribut krivulje koji sadrži sve točke i načine povezivanja. Instrukcije povezivanja su opisane jednim slovom prije brojčane vrijednosti (npr. M100,100). Ako je slovo veliko, brojka označava apsolutnu poziciju na platnu, a ako je slovo malo, brojka označava relativni pomak od trenutne točke. Slovo označava način povezivanja. Podržani načini su: pomak bez iscrtavanja (najčešće služi za pomak prve točke), ravna linija, horizontalna i vertikalna linija, kubna Bezierova krivulja, kvadratna Bezierova krivulja, eliptični luk i zatvaranje krivulje s početnom točkom.

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink">

    <path d="M100,50 L20,310 200,200 350,100
        400,300 430,20 500,300" fill="none"
        stroke="grey" stroke-width="2" />
    <path d="M100,50 Q20,310 200,200 350,100
        400,300 430,20 500,300" fill="none"
        stroke="orange" stroke-width="4" />
</svg>
```

Primjer 3: Dvije krivulje s identičnim koordinatama osim što se u atributu "d" razlikuju u slovu L (line) i Q (quadratic Bezier) koje određuju način povezivanja. (Curves.svg)

Slika 2: Prikaz primjera 3.

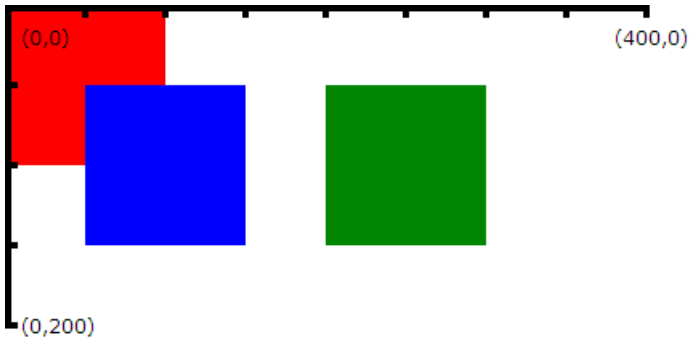
SVG platno opisuje prostor gdje se iscrtava sadržaj. To platno je beskonačno u veličini u svim dimenzijama, no iscrtavanje se ipak odvija u nekom pravokutnom prostoru koji se naziva prozor. Taj prozor može biti implicitno određen veličinom oblika ili može biti ekspilictno zadan u prvom `<svg>` elementu ili u vanjskom dokumentu koji sadrži cijeli SVG dokument.

Prilikom inicijalizacije prozora koordinatni sustav prozora i korisnički koordinatni sustav su jednaki. Ishodište oba sustava je u gornjem lijevom uglu prozora, a vrijednost koordinate se pozitivno povećava prema desno i

prema dolje. Novi koordinatni sustav se može inicijalizirati korištenjem novog <svg> elementa koji u sebi može sadržavati informacije o novom položaju. Svi elementi unutar tog <svg> elementa će računati poziciju prema njemu, a ne prema osnovnom koordinatnom sustavu prozora. Korisnički koordinatni sustav se mijenja nakon prve transformacije elementa. Na svaki element se može primjeniti više transformacija i to onim redom kako su ugniježđene. Podržane transformacije su: translacija, rotacija, skaliranje – promjena mjerila i smik – uzdužna transformacija. Kompozicija transformacija je zapravo preslikavanje korisničkih koordinata elementa u koordinate SVG prozora.

```
...
<svg x="0" y="0">
    <rect x="0" y="0"
        width="100" height="100" fill="red" />

    <svg x="50" y="50">
        <rect x="0" y="0"
            width="100" height="100" fill="blue" />
    </svg>
</svg>

<g transform="translate(200,50)">
    <rect x="0" y="0"
        width="100" height="100" fill="green" />
</g>
...
```

Primjer 4: Koordinatni sustav prvog <svg> elementa je identičan sustavu prozora. Drugi <svg> element definira novi koordinatni sustav od pozicije (50,50). Iako je pozicija obuhvaćenog kvadrata (0,0) on će biti iscrtan na poziciji (50,50) cjelokupnog prozora. Istu stvar nije moguće napraviti <g> elementom, no translacijom se mijenja korisnički koordinatni sustav. Kvadrat unutar <g> elementa također ima definiranu poziciju na (0,0), no iscrtan je na mjestu gdje počinje novi korisnički kooridnatni sustav. (NewCoordinate.svg)



Slika 3: Prikaz primjera 4.

2.3. Animiranje SVG-a

Svi statički elementi u SVG-u imaju mogućnost animacije. Dva su načina animacije podržani specifikacijom. Jedan od njih je deklarativni način pomoću SMIL-a.

SMIL (Synchronized Multimedia Integration Language) je jezik baziran na XML-u kojim se integrira i upravlja multimedijalnim sadržajima kao što su video, audio, tekst i grafika na Webu. Neki od najvažnijih mogućnosti SMIL-a su specifikacija rasporeda elemenata i sinkronizacija, prilagodba protoku informacija, rezoluciji ekrana i dubini boje. No iako SMIL postoji već 10 godina, njegova upotreba je vrlo rijetka i zapravo ne postoji neka veća ili vrlo raširena aplikacija ili tehnologija koja ga koristi. SMIL je imao malog doticaja s multimedijalnim porukama u mobilnoj telefoniji (MMS) i nedavno s HD DVD-om, no kako je Toshiba početkom 2008. godine objavila da prestaje s proizvodnjom i podrškom tog formata u korist Blu-raya pitanje je kako će se SMIL razvijati.

SMIL verzija 3.0 je trenutni predložak W3C-a. Od verzije 2.0 SMIL podržava ugradnju u druge XML jezike, pa tako i SVG. Unutar svakog SVG elementa se mogu deklarativno opisati manipulacije nad tim elementom. Četiri elementa za animaciju koji su definirani u SMIL specifikaciji su: `<animate>` (skalarnim vrijednostima se mogu dodijeliti različite vrijednosti u vremenu), `<set>` (slično kao i animate, samo za ne-numeričke vrijednosti),

`<animateMotion>` (pomiče objekt po putanji) i `<animateColor>` (mijenja boju objekata u vremenu). Kombinacijom tih deklarativnih naredbi u pravilnom redoslijedu je moguće postići vrlo kompleksne animacije.

Zbog slabe podrške SMIL-a u web preglednicima razvojni se inženjeri okreću imperativnoj animaciji pomoću ECMAScripta. ECMAScript je interpretirani jezik koji je najčešće znan kao JavaScript (dalje u tekstu će se koristiti ovo ime) ili Jscript po svojim najpoznatijim dijalektima. JavaScript je stekao ogromnu popularnost u vrlo kratko vrijeme i sada je dominatni jezik za skriptiranje klijentske strane na Webu. Podržan je od svih poznatih web preglednika, a odnedavno se mogu naći i web preglednici na mobilnim uređajima koji ga podržavaju. Upravo su standardiziranost i raširenost JavaScripta doprinijeli korištenju istog za animiranje, interaktivnost ili bilo kakvu manipulaciju SVG-a.

Da bi JavaScript mogao dinamički mijenjati neki sadržaj on mora imati neku predodžbu o njegovom izgledu. Dugo vremena su web preglednici prezentirali HTML svatko na svoj način i zato se JavaScript morao prepisivati za svaki preglednik. Da bi se to sučelje standardiziralo, stvoren je DOM. DOM (Document Object Model) je objektni model neovisan o platformi i jeziku za prikaz HTML-a, XML-a i sličnih formata. Ti formati su hijerarhijski i mogu se opisati stablom. Web preglednici nisu dužni prikazivati HTML pomoću DOM-a, no u želji za kompatibilnošću većina najpoznatijih preglednika se većim dijelom ili potpuno oslanja na DOM.

Opisivati sve mogućnosti JavaScripta bi bilo nemoguće, no dovoljno je reći da JavaScript kroz DOM može doći do svakog dijela SVG dokumenta i po želji ga promijeniti. Krajnost toga je da se cijeli SVG dokument može kreirati dinamički na klijentskoj strani. Za razliku od deklarativnog SMIL-a, JavaScript je imperativan, pa je nužno čuvati stanje i računati svaku željenu promjenu.

3. Web aplikacije

U ranijim tipovima klijent-poslužitelj načinima komunikacije svaka je aplikacija imala svoj klijentski program koji je služio kao korisničko sučelje i koji se morao instalirati na svako računalo. Većina je promjena na aplikaciji zahtijevala promjenu na klijentskoj strani što je povećavalo troškove podrške i smanjivalo produktivnost. Klijentska je aplikacija morala biti drugačija za svaki operacijski sustav. Proizvođači aplikacije su se vrlo često odlučili za jednu tehnologiju i jedan operacijski sustav i time je dijelu korisnika bilo onemogućeno ili jako otežano njeno korištenje.

Web aplikacija je također tipa klijent-poslužitelj, no kao svoj klijent koristi web preglednik. Sve web aplikacije se strukturiraju kao niz HTML dokumenata. Kako je HTML tehnologija nevezana uz operacijski sustav, svatko može koristiti web aplikaciju bez potrebe za instaliranjem posebnih klijenata. Razvojnim inženjerima to jako olakšava pokretanje, podršku i poboljšavanje aplikacije. Jedino čega se moraju držati su propisani standardi tehnologija koje podržava web preglednik jer im on postaje nova razvojna platforma. Neke od tih tehnologija su HTML, XML, CSS i JavaScript.

U početku su web aplikacije bile samo niz statičkih HTML dokumenata. Oni su se prenosili preko HTTP-a koji je zamišljen kao protokol koji ne čuva stanje, a time ni stanje klijenta. Stoga su aplikacije bile vrlo jednostavne. Da bi povećali interaktivnost i korisnost web aplikacija razvojni su inženjeri napravili nekoliko velikih koraka.

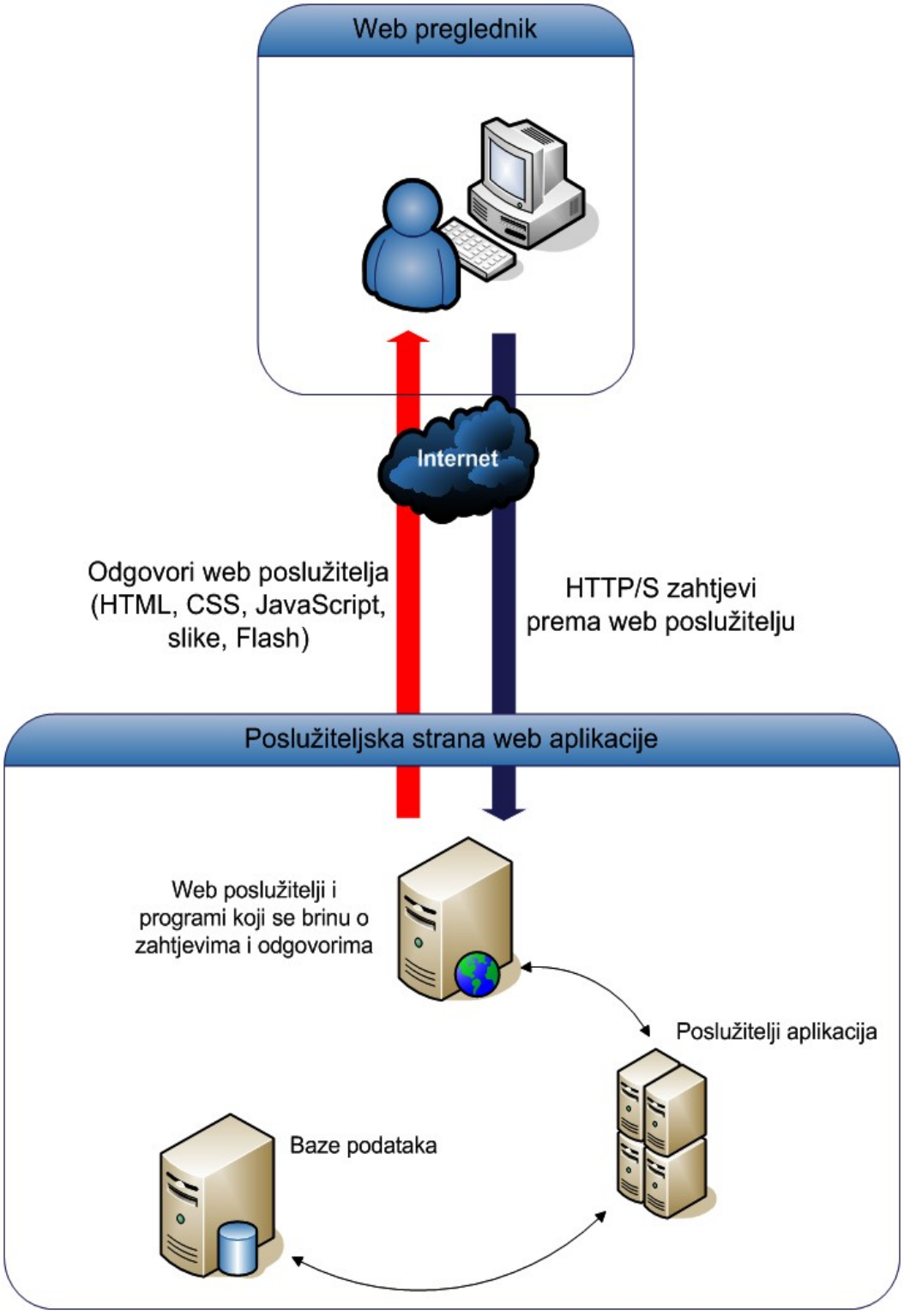
- Stanje aplikacije ili klijenta se počelo zapisivati unutar HTML dokumenata ili se slalo web pregledniku u obliku internet kolačića. Na taj se način moglo na serverskoj strani odrediti gdje se klijent nalazi, koje su mu mogućnosti, ovlasti, itd.
- Klijentu više nisu posluživani samo statički HTML dokumenti, nego se dokument dinamički stvarao na strani poslužitalja ovisno o ulazu i željama klijenta. Neke od poznatijih tehnologija koje to omogućuju su

ASP, ASP.NET, CGI, JSP/Java, PHP, Perl, Python i Ruby on Rails. Sve one na sebi svojstven način prihvaćaju HTTP zahtjev od klijenta, analiziraju ga i ovisno o stanju zapisanom u HTML dokumentu stvaraju novi koji šalju kao odgovor na zahtjev. Zbog tih su tehnologija postale moguće web aplikacije poput web elektroničke pošte, internet dućana, foruma, blogova i sl.

- Skriptiranjem na klijentskoj strani je povećana i ubrzana interaktivnost korisnika i aplikacije. Do tog trenutka je za svaku promjenu na ekranu trebalo poslati zahtjev poslužitelju koji bi zatim vratio drugačiji dokument. Skriptiranjem su se unutar web preglednika mogli obaviti neki zadaci za koje onda nije bilo potrebno kontaktirati poslužitelj. Razvojem tehnologije AJAX postalo je moguće poslati zahtjev bez ponovnog prikazivanja cijele stranice. Tako su web aplikacije sve više počele nalikovati na klasične.

No i web aplikacije imaju svojih mana. Web preglednik je jedna razvojna platforma, no ne implementiraju svi web preglednici sve standarde u potpunosti ili točno kako je propisano. Male razlike u ponašanju i prikazu elemenata cijelo vrijeme prisiljavaju razvojne inženjere da odvaguju tehnološki napredak i broj korisnika. Većina korisnika preferira jedan web preglednik i radije neće koristiti web aplikaciju ako to podrazumijeva prelazak na neki drugi. Da ne bi ostali bez tih korisnika, razvoj se uglavnom uvijek vrši na samo provjerenim i ustaljenim tehnologijama.

Kako su sve web aplikacije smještene na poslužitelju, a klijent koristi web preglednik samo da bi im pristupio, nedostatkom veze do poslužitelja korisnik osim što ne može pristupiti samoj aplikaciji ne može ni pristupiti svojim informacijama i stanju (npr. elektronička pošta, povijest bankovnih transakcija, stanje pošiljke).



Slika 4: Dijagram uobičajene komunikacije između korisnika i web aplikacije.

4. Programsko ostvarenje

4.1. Ideja i tehnologija

Na Internetu se počelo pojavljivati jako puno aplikacija i servisa koji su donedavno bili vezani isključivo za stolna računala i njihove instalirane programe. Neki od njih su kalkulatori, uređivači slika, tekst procesori i pohrana podataka. Ovaj se projekt bavi izradom grafova. Iako već postoji nekoliko sličnih programa ovaj je specifičan po tome što je izlaz u SVG formatu, a sučelje je vrlo jednostavno za korištenje. Grafika na Internetu se razvija presporo zbog inercije koju je nametnula rasterska grafika, ali i nedostatka educiranosti i alata kojim bi se taj razvoj ubrzao. Ideja cijelog projekta jest ponuditi jedan jednostavan alat u novoj i obećavajućoj tehnologiji.

Alat je u obliku web aplikacije. Napisana je u programskom jeziku C# u tehnologiji ASP.NET. Aplikaciju se može smatrati i servisom jer nije potrebno nikakvo grafičko sučelje kako bi se došlo do rezultata. Potrebno je sve željene podatke o grafu upisati u *query string* u adresi željenog resursa (URL) na što web preglednik šalje HTTP GET zahtjev koji aplikacija obrađuje i vraća graf u SVG-u.

4.2. Struktura i način rada aplikacije

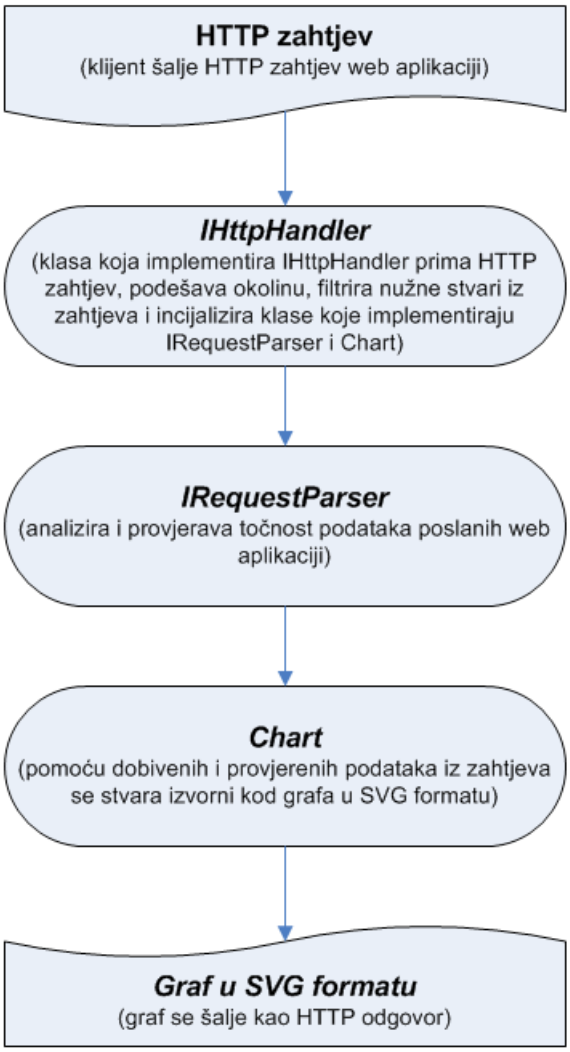
Aplikacija se sastoji od nekoliko datkoteka koje čine izvorni kod i konfiguracijske datoteke (Web.Config). Ona je specifična za ovu vrstu tehnologije, ali u ovoj aplikaciji nema neku posebnu namjenu. Datoteke izvornog koda su sljedeće:

Naziv datoteke	Objašnjenje
ChartGen.ashx	Sadrži klasu koja implementira sučelje IHttpHandler. Ta se klasa brine za uređivanje okoline, inicijalizacije svih većine potrebnih objekata i tok jednog prolaska kroz aplikaciju.
BarChart.cs	Sadrži klasu koja nasljeđuje apstraktnu klasu Chart. Klasa BarChart se brine za sve specifičnosti stupčastog grafa.
Chart.cs	Sadrži vršnu klasu grafova koja se brine o strukturi izvornog koda SVG grafa, sadrži sve funkcije i varijable svojstvene svim grafovima, brine o pravilnom redoslijedu izvođenja funkcija i delegira iscrtavanje specifičnih dijelova grafa na ostale klase.
ChartException.cs	Sadrži aplikacijsku iznimku koja se pojavljuje u slučaju greške.
ChartInformation.cs	Sadrži klasu koja čuva sve podatke po kojima klasa Chart izgrađuje graf.
Enums.cs	Sadrži sve enumerirane tipove podataka u aplikaciji.
Histogram.cs	Sadrži klasu koja nasljeđuje apstraktnu klasu Chart. Klasa Histogram se brine za sve specifičnosti histograma.
IRequestParser.cs	Sadrži sučelje koje analizira i provjerava podatke dobivene iz zahtjeva i sprema ih u objekt klase ChartInformation.
LineChart.cs	Sadrži klasu koja nasljeđuje apstraktnu klasu Chart. Klasa LineChart se brine za sve specifičnosti linijskog grafa.
Margins.cs	Sadrži klasu koja čuva informacije o marginama unutar grafa.

PieChart.cs	Sadrži klasu koja nasljeđuje apstraktnu klasu Chart. Klasa PieChart se brine za sve specifičnosti kružnog grafa.
QueryStringRequestParser.cs	Sadrži klasu koja implementira sučelje IRequestParser i pretvara informacije iz *query stringa* u valjani objekt klase ChartInformation.
SimpleChartFactory.cs	Sadrži klasu koja inicijalizira objekte grafova.

Tablica 1: Popis datoteka aplikacije.

Aplikacije je protočnog tipa. To znači da aplikacija ima točno definirani ulaz, protočni sustav koji slijedno obrađuje ulazne informacije i izlaz. Nakon što korisnik pošalje zahtjev on nema nikakvog pristupa samoj aplikaciji niti ne može utjecati na njen rad. Takva je arhitektura vrlo česta u programima koji analiziraju ili transformiraju podatke.



Slika 5: Tok rada aplikacije.

4.3. Ulazni podaci

Ulazni podaci se šalju aplikaciji unutar *query stringa*. *Query string* je niz znakova u adresi resursa nakon znaka "?". U njemu se mogu poslati podaci koje web aplikacija može iskoristiti za svoj rad. Podaci su oblika `ime=vrijednost`, a odvojeni su znakom "&". Podržana duljina *query stringa* je bila vrlo mala, no moderni web preglednici omogućavaju najmanje 2000 znakova, ovisno o implementaciji. Ta je količina sasvim dovoljna kako bi podržala normalno korištenje ove aplikacije.

U primjeru

`http://www.google.hr/search?hl=hr&q=fer`

`http` označava protokol, `www.google.hr/search` je adresa resursa, a `?hl=hr&q=fer` je *query string* koji kaže da je vrijednost parametra `hl` (jezik pretraživanja) jednaka `hr`, a vrijednost parametra `q` (pojam koji se pretražuje) jednaka `fer`.

Napravljena aplikacija podržava niz parametara od kojih su jedino podaci (data), veličina (size) i tip (type) grafa nužni, dok su ostali opcionalni.

Naziv	Opis parametra
type	Definira tip grafa. Moguće vrijednosti su: `line`, `bar`, `hist` i `pie`. Primjer: `type=bar`
size	Definira inicijalnu veličinu grafa. Iako je graf vektorski i veličina mu se može podešavati bez gubitka kvalitete, ova će vrijednost definirati početni odnos veličina kao što su margine, veličina slova, debljina linija. Vrijednost parametra je u formatu `ŠIRINAxVISINA`. Vrijednosti moraju biti cjelobrojne. Primjer: `size=600x400`
data	Podaci koji će se iscrtati u grafu. Svi se podaci zapisuju isto, no čitat će se ovisno o tipu odabranog grafa. Vrijednosti su brojčane i mogu biti decimalne, a za odvajanje cjelobrojnog i decimalnog dijela se koristi točka ("."). Zarezom (",") se međusobno odvajaju vrijednosti, a setovi vrijednosti znakom "|". Negativne vrijednosti nisu dozvoljene. Primjer: `data=1,4,1,2.3|2,3,1,3.4|1,2.9,10,15`
axes	Deklarira labele na osima grafa. Moguće vrijednosti su: `x` i `y`. Mogu se deklarirati obje vrijednosti i tada se odvoje zarezom. Redoslijed deklaracije je bitan jer se tim redoslijedom čita parametar labels. Primjer: `axes=y,x`
labels	Vrijednosti labela koje će se pojaviti na osima grafa. Labele se odvajaju zarezom (","), a setovi znakom "|". Redoslijed setova labela ovisi o redoslijedu deklaracije u parametru axes. Primjer: `labels=0,25,50,75,100|` `pon,uto,sri,cet,pet,sub,ned`

Tablica 2: Popis ulaznih parametara aplikacije.

4.4. Izlazni podaci

Aplikacija kao izlaz vraća HTTP odgovor MIME tipa `image/svg+xml`. Na taj način web preglednik prepoznaje tip sadržaja i zna kako ga treba prikazati. Sadržaj HTTP odgovora je SVG zapis grafa. S dobivenim grafom se može raditi bilo što, npr. umetnuti u neki web dokument, no zanimljivo je da se u taj isti web dokument može postaviti adresa aplikacije s query stringom koji sadrži informacije od tom grafu. Na taj način nije potrebno spremati izvorni kod grafa, nego se on svaki put stvori prilikom zatraživanja spomenutog web dokumenta. Takva je mogućnost posebno zanimljiva ako se web dokument i pripadni query string stvaraju dinamički na poslužiteljskoj strani prilikom zahtjeva. Primjer bi mogla biti financijska web aplikacija koja prikazuje izvješća s burze. Ta bi aplikacija mogla sakupljati informacije o dnevnoj trgovini dionicama, no nema mogućnost crtanja istih. U tom bi slučaju financijska aplikacija dinamički koristila usluge napravljene aplikacije za SVG grafove.

```
...
<h1>Graf današnjeg trgovanja</h1>
<object data="http://www.financije.hr/chartgen.ashx?
type=line&size=600x400&data=1,4,1,2.3,2,3,1,3.4,1,2.9,
7,3,4,2,8,7,3,4,4,5" type="image/svg+xml"
 width="600" height="400">
</object>
<p>...
```

Primjer 5: Jedan od načina kako bi se web aplikacija za SVG grafove mogla dinamički pozivati iz web dokumenata.

Slika 6: Ovako bi se prikazao graf pozvan iz primjera 5.

Kako su svi grafovi naslijeđeni iz jedne klase koja se brine za strukturu i redoslijed, izvorni kod svih grafova je vrlo sličan. Ispis izvornog koda primjera 5 i slike 6 se može vidjeti u primjeru 6. Nakon deklaracije korijenskog elementa `<svg>` dolaze elementi `<style>` i `<defs>`.

Unutar `<style>` se definiraju CSS klase koje se primjenjuju niže u SVG zapisu. Na taj se način centralizira mjesto gdje je opisan dio izgleda grafičkih elemenata (boja, transparentnost, širina i boja ruba, itd.) kako bi kasnije bilo lakše modificirati graf. Isto tako se smanjuje veličina zapisa jer je u ovakvom slučaju potrebno samo jednom opisati izgled elementa, a zatim se taj opis samo dodijeli svim potrebnim elementima u obliku CSS klasa.

`<defs>` element je vrlo sličan elementu `<style>`. On služi za deklaraciju čitavih elemenata ili čak grupa elemenata. Deklaracijom unutar `<defs>` ti se elementi ne prikazuju, no kasnije ih je moguće pomoću XLink tehnologije koristiti više puta tako što se referencira njihov jedinstveni atribut "`id`". Beneficije toga su iste kao i u `<style>` elementu, jednostavnija modifikacija i smanjena veličina zapisa.

Sljedećim `<svg>` elementom se otvara novo platno na koje se iscrtava sam graf – linije, stupci ili kružni isječci. Veličina i položaj tog platna se računaju ovisno o postojanju i veličini labela.

Nakon njega dolaze preostali elementi koji iscrtavaju osi i labele.

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
width="600" height="400">
     <style type="text/css">
     .line {fill:none; stroke-width:3;}
     .axis {stroke:grey; fill:none; stroke-width:2;
     stroke-linecap:square;}
     </style>

     <defs></defs>

     <svg x="10" y="10" width="580" height="380">
     <polyline style="stroke:skyblue;" class="line"
points="0,333 30,190 60,333 90,271 120,285 150,238
180,333 210,219 240,333 270,243 300,48 330,238
360,190 390,285 420,0 450,48 480,238 510,190 540,190
570,143 " />
     </svg>

     <polyline class="axis" points="10,10 10,390
590,390 "/>
</svg>
```

Primjer 6: Izvorni kod jednostavnog grafa.

4.5. Linijski graf

Linijski graf sa svim opcijama moguće je dobiti zahtjevom

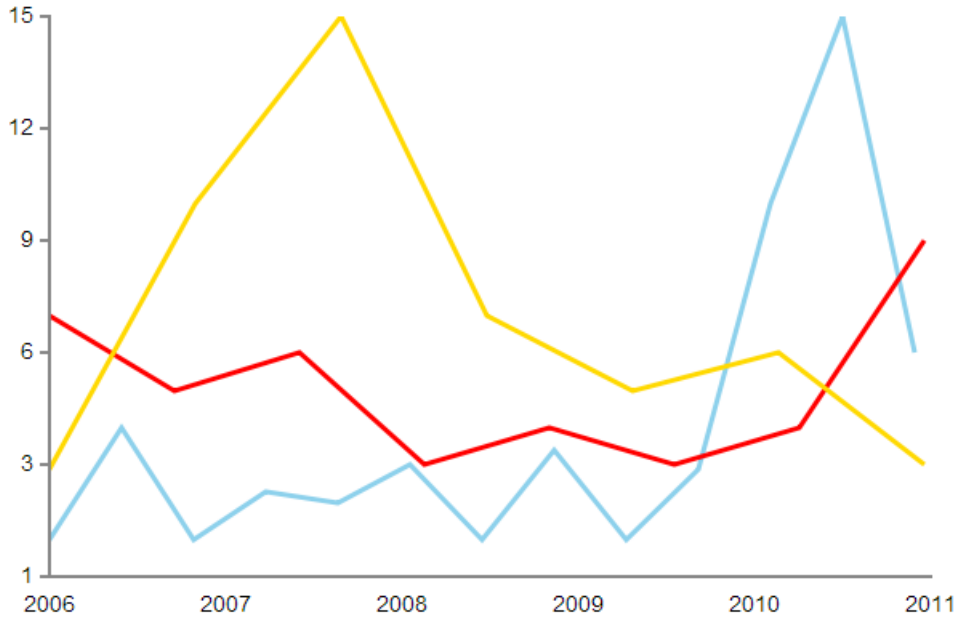
```
http://neka.domena.hr/chartgen.ashx?
type=line&size=600x400&data=1,4,1,2.3,2,3,1,3.4,1,2.9,10,
15,6|7,5,6,3,4,3,4,9|
2.9,10,15,7,5,6,3&axes=x,y&labels=2006,2007,2008,2009,201
0,2011|1,3,6,9,12,15
```

Linijski graf podržava više linija koje ne moraju imati isti broj vrijednosti u setu. Na taj način je povećana fleksibilnost, no moguća je mala narušenost točnog pregleda vrijednosti. Labele se ravnomjerno raspoređuju na obje osi ovisno o njihovom broju i ne moraju biti jednakog broja kao i neki broj vrijednosti u setu.

Izvorni kod linijskog grafa moguće je pogledati na primjeru 7, a prikaz na slici 7.

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" width="600"
height="400">
<style type="text/css">
.line {fill:none; stroke-width:3;}
.axis {stroke:grey; fill:none; stroke-width:2; stroke-
linecap:square;}
.labels {font-family:Arial,sans-serif; font-
size:14px;}
</style>

<defs>
<line id="labelMarker" class="axis" x1="-5" y1="0"
x2="0" y2="0" />
</defs>

<svg x="30" y="20" width="550" height="350">
<polyline style="stroke:skyblue;" class="line"
points="0,327 45,257 90,327 135,297 180,304 225,280
270,327 315,271 360,327 405,283 450,117 495,0 540,210
" />
<polyline style="stroke:red;" class="line"
points="0,187 78,234 156,210 234,280 312,257 390,280
468,257 546,140 " />
<polyline style="stroke:gold;" class="line"
points="0,283 91,117 182,0 273,187 364,234 455,210
546,280 " />
</svg>

<polyline class="axis" points="30,20 30,370 580,370 "/
>
<text text-anchor="middle" x="30" y="392"
class="labels">2006</text>
...

<text text-anchor="end" x="20" y="374.2"
class="labels">1</text>
<use xlink:href="#labelMarker" x="30" y="370" />
...

</svg>
```

Primjer 7: Izvorni kod potpunog linijskog grafa. Dijelovi koji se ponavljaju su zamijenjeni s "...". (FullMultilineChart.svg)



Slika 7: Prikaz potpunog linijskog grafa iz primjera 7.

4.6. Stupčasti graf

Stupčasti graf sa svim opcijama moguće je dobiti zahtjevom

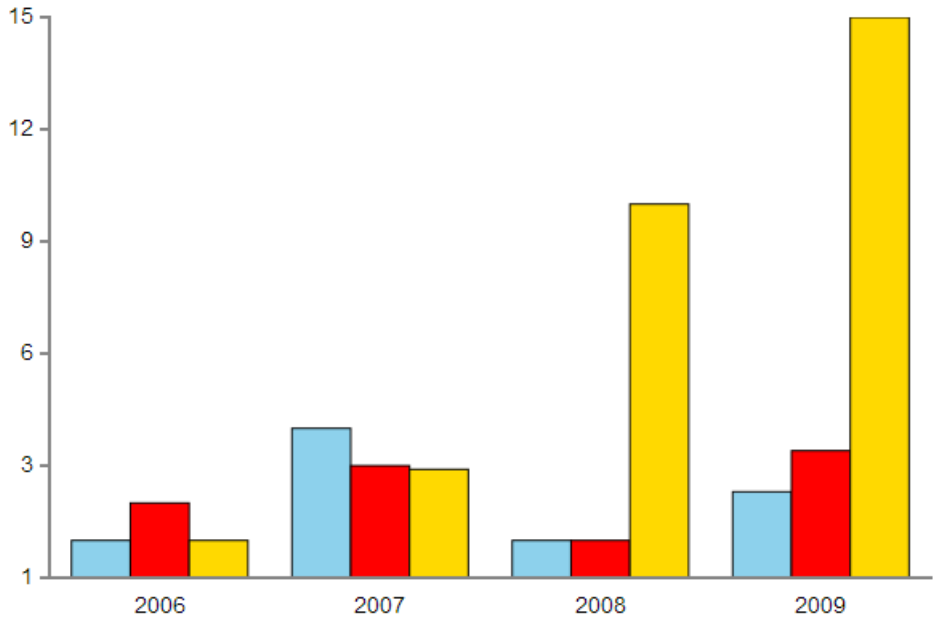
```
http://neka.domena.hr/chartgen.ashx?
type=bar&size=600x400&data=1,4,1,2.3|2,3,1,3.4|
1,2.9,10,15&axes=y,x&labels=1,3,6,9,12,15|
2006,2007,2008,2009
```

Stupčastim se grafom može iscrtati samo jedan stupac ili veći broj setova i stupaca u svakome. Broj stupaca mora biti jednak u svim setovima, a broj labela na apscisi jednak broju setova.

Izvorni kod linijskog grafa moguće je pogledati na primjeru 8, a prikaz na slici 8.

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" width="600"
height="400">
<style type="text/css">
.bar0 {fill:skyblue; stroke-width:1; stroke:black;}
.bar1 {fill:red; stroke-width:1; stroke:black;}
.bar2 {fill:gold; stroke-width:1; stroke:black;}
.axis {stroke:grey; fill:none; stroke-width:2; stroke-
linecap:square;}
.labels {font-family:Arial,sans-serif; font-
size:14px;}
</style>
<defs>
<line id="labelMarker" class="axis" x1="-5" y1="0"
x2="0" y2="0" />
</defs>
<svg x="30" y="20" width="550" height="350">
<rect class="bar0" x="13.75" y="326.7" width="36.7"
height="23.3" />
...
<rect class="bar1" x="50.45" y="303.3" width="36.7"
height="46.7" />
...
<rect class="bar2" x="87.15" y="326.7" width="36.7"
height="23.3" />
...
</svg>

<polyline class="axis" points="30,20 30,370 580,370 "/
>
<text text-anchor="end" x="20" y="374.2"
class="labels">1</text>
...
<text text-anchor="middle" x="98.75" y="392"
class="labels">2006</text>
...
</svg>
```

Primjer 8: Izvorni kod potpunog linijskog grafa. Dijelovi koji se ponavljaju su zamijenjeni s "...". (FullBarChart.svg)



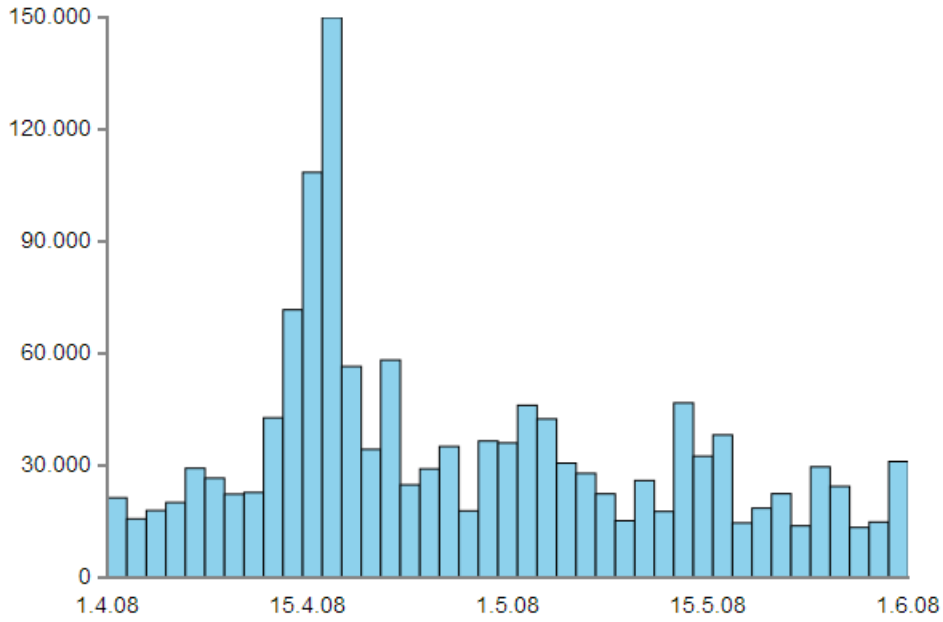
Slika 8: Prikaz potpunog stupčastog grafa iz primjera 8.

4.7. Histogram

Histogram je moguće dobiti zahtjevom

```
http://neka.domena.hr/chartgen.ashx?
type=hist&size=600x400&data=21785,16042,18326,20509,29877
,27152,22799,23277,43671,73179,110790,153124,57727,35028,
59477,25394,29685,35812,18251,37317,36814,47062,43355,312
93,28499,22883,15519,26511,18080,47732,33178,38994,14936,
18984,22909,14123,30272,24942,13674,15203,31699&axes=x,y&
labels=1.4.08,15.4.08,1.5.08,15.5.08,1.6.08|
0,30.000,60.000,90.000,120.000,150.000
```

Izvorni kod histograma je skoro identičan onome od stupčastog grafa osim što se uzimaju vrijednosti iz samo prvog seta zadanih vrijednosti i iscrtavaju se bez razmaka.



Slika 9: Prikaz histograma. (Histogram.svg)

4.8. Kružni graf

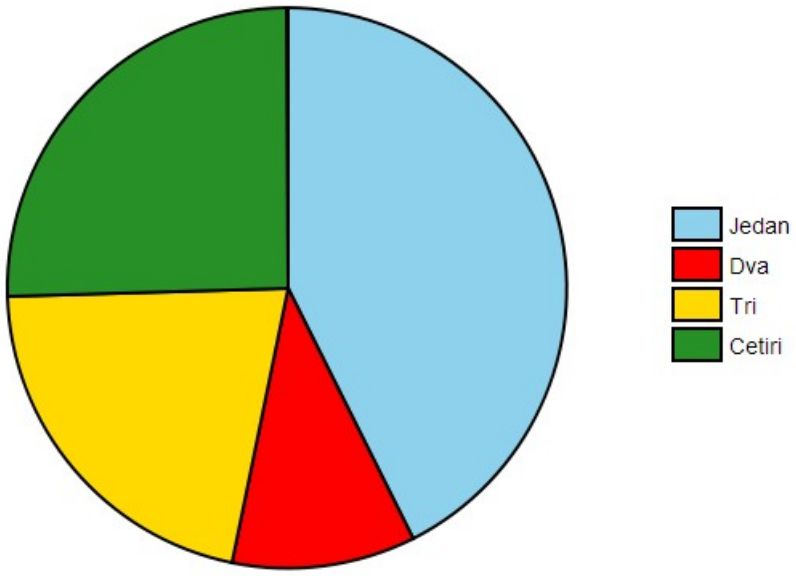
Kružni graf je moguće dobiti zahtjevom

```
http://localhost/svgcharts/chartgen.ashx?
type=pie&size=600x400&data=4,1,2,2.4&labels=Jedan,Dva,Tri
,Cetiri
```

Kružni isječak računa se iz vrijednosti samo prvog seta. Ne postoje osi i labele, no zato se u parametar `labels` upisuju vrijednosti legende.

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
width="600" height="400">
<style type="text/css">
.piesegment {stroke:black; stroke-width:2;}
</style>
<svg x="20" y="20" width="480" height="360">
<path d="M240,180 L240,5 A175,175 0 0,1 318,336 z"
fill="skyblue" class="piesegment" />
<path d="M240,180 L318,336 A175,175 0 0,1 205,351 z"
fill="red" class="piesegment" />
<path d="M240,180 L205,351 A175,175 0 0,1 65,185 z"
fill="gold" class="piesegment" />
<path d="M240,180 L65,185 A175,175 0 0,1 239,5 z"
fill="forestgreen" class="piesegment" />
</svg>

<rect x="500" y="150" width="30" height="20"
stroke="black" stroke-width="2" fill="skyblue"/>
<text x="535" y="166" style="font-family:Arial,sans-
serif; font-size:14px;">Jedan</text>
...
</svg>
```

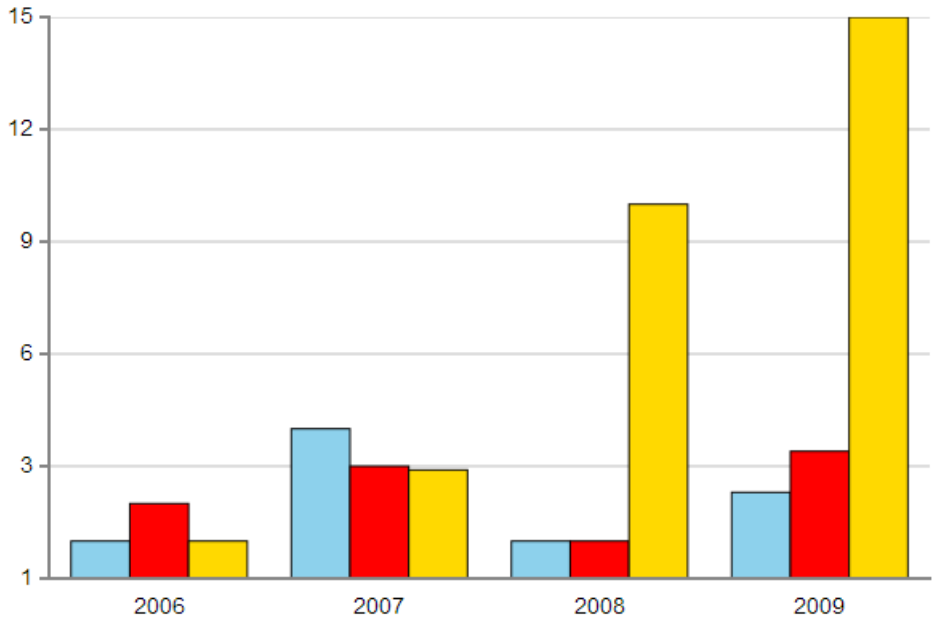
Primjer 9: Izvorni kod kružnog grafa. (FullPieChart.svg)



Slika 10: Prikaz primjera 9.

4.9. Dodatne mogućnosti

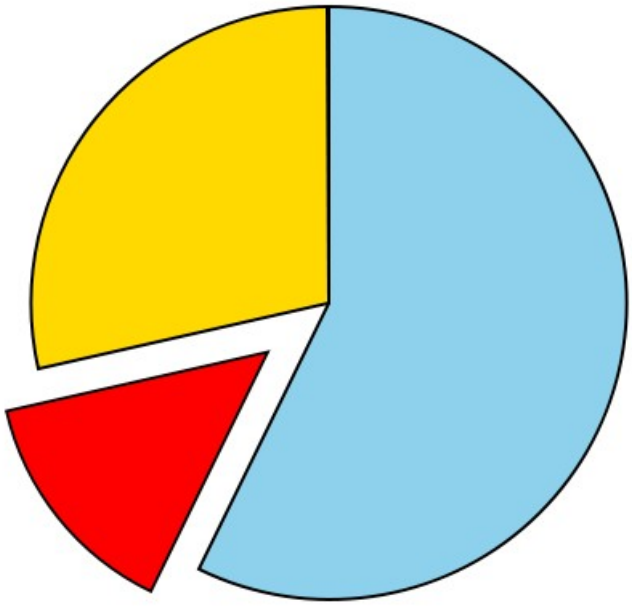
Kako je svaki SVG zapis grafa čitljiv čovjeku tako je vrlo lako moguće promijeniti graf dobiven iz web aplikacije. Sve što je potrebno jest običan uređivač teksta i zapis grafa. U sljedećih nekoliko slika i primjera pokazano je par načina dodatne dorade grafova.



Slika 11: Dodavanjem nekoliko jednostavnih linija vrlo je lako poboljšati preglednost i atraktivnost grafa. (FullBarChartGrid.svg)

```
...
<svg x="10" y="10" width="580" height="380">
<path d="M290,190 L290,5 A185,185 0 1,1 209,356 z"
fill="skyblue" class="piesegment" />
<path transform="scale(0.90),translate(-10,55)"
d="M290,190 L209,356 A185,185 0 0,1 109,231 z"
fill="red" class="piesegment" />
<path d="M290,190 L109,231 A185,185 0 0,1 289,5 z"
fill="gold" class="piesegment" />
</svg>
...
```

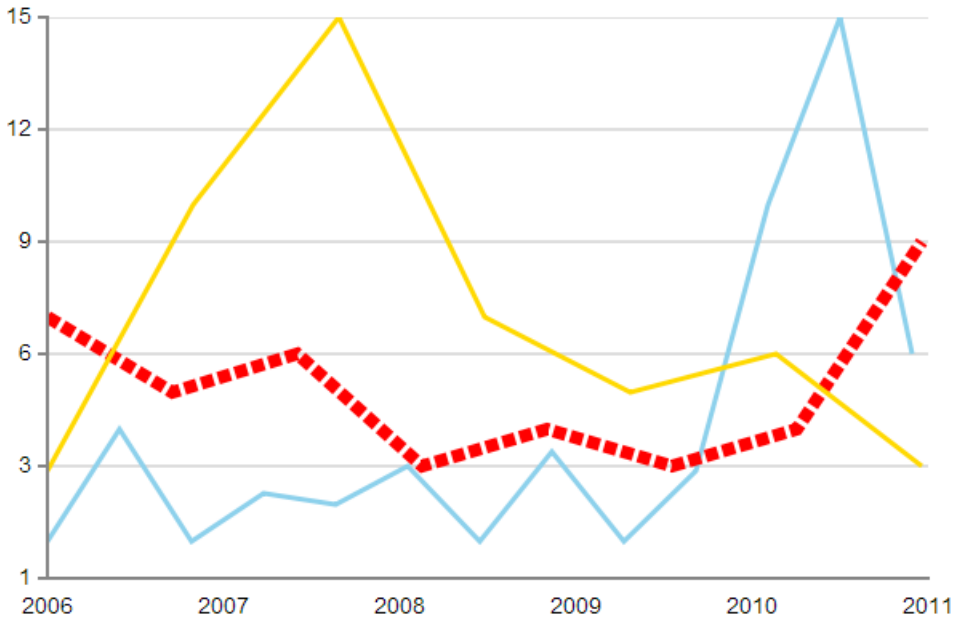
Primjer 10: Na jedan isječak kružnog grafa primjenjene su transformacije skaliranja i translacije kako bi se dodatno označio. (AlteredBarePieChart.svg)



Slika 12: Prikaz primjera 10.

```
...
<style type="text/css">
.line {fill:none; stroke-width:3;}
.line:hover {stroke-dasharray:8,3; stroke-width:8;}
...
</style>
...
```

Primjer 11: Jednostavnim dodavanjem pseudo-klase "hover" u CSS klasi linije grafa se istaknu kada se pokazivačem prelazi preko njih. (AlteredFullMultilineChartGrid.svg)



Slika 13: Prikaz primjera 11. U ovom se slučaju pokazivač nalazi iznad crvene linije.

5. Zaključak

Uvjeti diplomskog rada su proučiti mogućnosti SVG formata i razraditi web aplikaciju za dinamičku izradu grafova i animiranih objekata.

Izvedeno programsko rješenje pokriva sve uvjete diplomskog rada, neke u potpunosti, a neke samo u određenoj mjeri. Web aplikacija koristi HTTP protokol za rad. Podržan je HTTP GET koji je za trenutne mogućnosti sasvim dovoljan i vrlo jednostavan za korištenje. Sučelje aplikacije je također pojednostavljeno kako se u čestom korištenju ne bi trebala previše konzultirati dokumentacija što je velika pomoć prilikom korištenja ovakvih servisa.

Izlaz aplikacije je graf u SVG zapisu koji se može dinamički stvarati unutar HTML stranice ili se može spremiti i onda koristiti na svim mjestima gdje je taj format podržan. Kako je SVG zapis tekstualan i otvoren, dobiveni graf se može vrlo lako mijenjati i prilagoditi različitim potrebama.

Iako je ostavljena mogućnost jednostavne interaktivnosti objekata grafa preko CSS-a, animacija nije implementirana u aplikaciji. Jednostavniji SMIL je još uvijek preslabo podržan i većina preglednika ga ne može nativno prikazati. Animacija JavaScriptom je bitno drugačija tehnologija od one kojom je implementirana sama aplikacija na poslužiteljskoj strani, pa bi za njenu realizaciju bilo potrebno puno više vremena od onog predviđenog za ovaj rad.

6. Sažetak / Abstract

Programska podrška za dvodimenzijsku grafiku na Web-u

U ovom radu je kratko opisan Scalable Vector Graphics (SVG) format i način rada web aplikacija. Programski je implementiran web servis koji na HTTP GET zahtjev vraća graf u SVG zapisu. Podržani grafovi su linijski, stupčasti, kružni i histogram. Tehnologija implementacije je ASP.NET, a korišteni programski jezik C#. Na gotovim grafovima su demonstrirane mogućnosti daljnje obrade zapisa.

Ključne riječi: Internet, Web, dvodimenzijska grafika, vektorska grafika, web aplikacija, SVG, graf

Application for two-dimensional web graphics

This paper describes Scalable Vector Graphics (SVG) format and how web applications work. Simple web service was created that receives HTTP GET reqest and returns chart in SVG. Supported charts are line chart, bar chart, pie chart and histogram. Technology in which this service is implemented is ASP.NET and C# programming language. Ways of customizing finished chart source code were demonstrated.

Key words: Internet, Web, two-dimensional graphic, vector graphic, web application, SVG, chart

7. Popis oznaka i kratica

SVG	tekstualno baziran vektorski format (Scalable Vector Graphics)
GIF, JPEG, PNG	neki od poznatijih rasterskih formata
W3C	međunarodna organizacija za standardizaciju Web tehnologija (World Wide Web Consortium)
PGML, VML	prethodnici SVG formata
XML	jezik za označavanje podataka (EXtensible Markup Language)
HTML (XHTML)	jezik za strukturiranje teksta u dokumentu (HyperText Markup Language)
CSS	jezik za formatiranje i prezentaciju (Cascading Style Sheets)
XSLT	dio XSL tehnologije za transformaciju XML dokumenata
XLink	jezik za označavanje referenci u XML dokumentima (XML Linking Language)
MMS	multimedijalne poruke u mobilnim komunikacijama (Multimedia Messaging Services)
SMIL	jezik kojim se integrira i upravlja multimedijalnim sadržajima (Synchronized Multimedia Integration Language)
DOM	objektni model za prikaz HTML-a, XML-a i sličnih formata (Document Object Model)
HTTP	komunikacijski protokol za razmjenu informacija na Internetu (Hypertext Transfer Protocol)
HTTP GET	tip zahtjeva u HTTP-u
URL	jedinstvena adresa resursa (Uniform Resource Locator)
query string	tekst u URL-u koji se nalazi iza znaka "?"
C#	objektno orijentirani programski jezik
ASP.NET	tehnologija za izradu web aplikacija
MIME	tip resursa (Multipurpose Internet Mail Extensions)

8. Popis tablica

Tablica 1: Popis datoteka aplikacije.....16
Tablica 2: Popis ulaznih parametara aplikacije.....19

9. Popis slika

Slika 1: Prikaz primjera 2 u web pregledniku s podrškom za SVG. Treba primjetiti redoslijed kojim su deklarirani elementi i redoslijed iscrtavanja istih. 5
Slika 2: Prikaz primjera 3. 7
Slika 3: Prikaz primjera 4. 9
Slika 4: Dijagram uobičajene komunikacije između korisnika i web aplikacije. 13
Slika 5: Tok rada aplikacije. 17
Slika 6: Ovako bi se prikazao graf pozvan iz primjera 5. 21
Slika 7: Prikaz potpunog linijskog grafa iz primjera 7. 25
Slika 8: Prikaz potpunog stupčastog grafa iz primjera 8. 27
Slika 9: Prikaz histograma. (Histogram.svg). 28
Slika 10: Prikaz primjera 9. 29
Slika 11: Dodavanjem nekoliko jednostavnih linija vrlo je lako poboljšati preglednost i atraktivnost grafa. (FullBarChartGrid.svg). 30
Slika 12: Prikaz primjera 10. 31
Slika 13: Prikaz primjera 11. U ovom se slučaju pokazivač nalazi iznad crvene linije. 32

10. Literatura

[1] SVG 1.1 specifikacija http://www.w3.org/TR/SVG11/

[2] Andrew H. Watt, *Designing SVG Web Graphics*, 2001

[3] Elliotte Rusty Harold, *XML 1.1 Bible, 3rd Edition*, 2004

[4] Mozilla SVG resursi http://www.croczilla.com/svg

[5] Kevin Lindsey Software Development - SVG i JavaScript poduke objavljeni na http://www.kevlindev.com/

[6] Wikipedia http://en.wikipedia.org/wiki/Svg