SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Krunoslav Žubrinić

PROGRAMSKA POTPORA STVARANJU OSOBNOG OKOLIŠA ZA UČENJE

MAGISTARSKI RAD

Zagreb, 2010.

Magistarski rad izrađen je na Zavodu za primijenjeno računarstvo
Fakulteta elektrotehnike i računarstva

Mentor: Prof.dr.sc. Damir Kalpić

Magistarski rad ima 134 stranice.

Rad br.: _____

POVJERENSTVO ZA OCJENU U SASTAVU:

1. Prof.dr.sc. Krešimir Fertalj – predsjednik
2. Prof.dr.sc. Damir Kalpić – mentor
3. Prof.dr.sc. Nataša Hoić-Božić – Odjel za informatiku Sveučilišta u Rijeci

POVJERENSTVO ZA OBRANU U SASTAVU:

1. Prof.dr.sc. Krešimir Fertalj – predsjednik
2. Prof.dr.sc. Damir Kalpić – mentor
3. Prof.dr.sc. Nataša Hoić-Božić – Odjel za informatiku Sveučilišta u Rijeci

Datum obrane: 15. siječnja 2010.g.

SADRŽAJ

1 Uvod 1
1.1 Motivacija za magistarski rad 1
1.2 Pregled dostignuća na polju razvoja osobnih okoliša za učenje 2
1.3 Cilj i struktura rada 4
2 Računalom podržano učenje 5
2.1 Obrazovanje i učenje 5
2.1.1 Podjela obrazovanja 5
2.1.2 Teorije učenja 6
2.2 Elektroničko učenje 8
2.2.1 Definicije pojma 8
2.2.2 Podjela elektroničkog učenja 9
2.2.3 Karakteristike elektroničkog učenja 9
2.2.4 Razvoj sustava za računalom podržano učenje. 10
2.3 Virtualno obrazovno okruženje 16
2.3.1 Definicija virtualnog obrazovnog okruženja 17
2.3.2 Karakteristike virtualnog obrazovnog okruženja 17
2.3.3 Opis virtualnog obrazovnog okruženja 18
2.3.4 Elementi virtualnog obrazovnog okruženja 18
3 Web 2.0 21
3.1 Web kao platforma 21
3.1.1 Razvoj Web 2.0 modela 21
3.1.2 Osnovne karakteristike Web 2.0 modela 22
3.1.3 Web servisi 22
3.1.4 Servisno orijentirana arhitektura 23
3.1.5 Resursno orijentirana arhitektura 25
3.1.6 Hibridne web aplikacije 28
3.1.7 *Widgeti* 28
3.1.8 RSS 28
3.2 Društvene mreže 30
3.2.1 Teorija društvenih mreža 30
3.2.2 Društvene mreže u Web 2.0 aplikacijama 30
3.2.3 Folksonomija 31
3.3 Web aplikacije s unaprijeđenim korisničkim sučeljem 31
3.3.1 Osobine RIA 32
3.3.2 RIA tehnologije 34
3.4 Utvrđivanje identiteta korisnika u Web 2.0 okruženju 36
3.4.1 Model decentraliziranog načina provjere identiteta 37
3.4.2 Potencijalni problemi u korištenju 38
3.4.3 Protokoli 39
3.5 Web 2.0 aplikacije 39

3.5.1 Elementi Web 2.0 aplikacija ... 39
3.5.2 Primjeri Web 2.0 aplikacija ... 40
4 Osobni okoliš za učenje ... 43
4.1 Potreba za razvojem osobnog okoliša za učenje ... 43
4.1.1 Odmak od tradicionalnog modela učenja ... 43
4.1.2 Konektivizam – teorija učenja u digitalnom dobu ... 43
4.1.3 Utjecaj tehnoloških promjena ... 44
4.2 Koncept osobnog okoliša za učenje ... 45
4.2.1 Definicija pojma ... 45
4.2.2 Osnovne karakteristike osobnog okoliša za učenje ... 46
4.2.3 Konceptualni model osobnog okoliša za učenje ... 47
4.2.4 Usporedba virtualnog obrazovnog okruženja i osobnog okoliša za učenje ... 48
4.3 Povijesni razvoj osobnih okoliša za učenje ... 49
4.3.1 Nastanak ideje ... 49
4.3.2 Utjecaj društvenih mreža na razvoj osobnih okoliša za učenje ... 50
4.3.3 Osobni okoliš za učenje baziran na Web 2.0 servisima ... 50
4.3.4 Osobni okoliši za učenje u obliku hibridnih web aplikacija ... 51
4.4 Usporedba sustava za stvaranje osobnog okoliša za učenje ... 51
4.4.1 Web 2.0 aplikacije kao agregatori sadržaja ... 51
4.4.2 mPLE ... 54
4.4.3 PLEF ... 56
4.4.4 Usporedba opisanih sustava za stvaranje PLE-a ... 57
5 Model aplikacije za podršku stvaranju osobnog okoliša za učenje ... 60
5.1 Opći zahtjevi ... 60
5.2 Opis potrebnih funkcionalnosti ... 60
5.3 Specifikacija zahtjeva ... 62
5.3.1 Korištena metodologija ... 62
5.3.2 Funkcionalni zahtjevi ... 62
5.3.3 Nefunkcionalni zahtjevi ... 68
5.4 Analiza specifičnih zahtjeva ... 70
5.4.1 Izrada konceptualne mape ... 70
5.4.2 Pohrana podataka ... 71
5.5 Model podataka ... 73
6 Realizacija aplikacije za podršku stvaranju osobnog okoliša za učenje ... 75
6.1 Opis aplikacije ... 75
6.2 Klijentski dio aplikacije ... 76
6.2.1 Korištena tehnologija ... 76
6.2.2 Struktura klijentskog dijela aplikacije ... 77
6.2.3 Prikaz podataka ... 78
6.3 Komunikacija između klijenta i poslužitelja ... 79
6.3.1 Način prijenosa podataka između klijenta i poslužitelja ... 79
6.3.2 Prijenos podataka o resursima ... 81

6.4 Poslužiteljski dio aplikacije 82
6.4.1 Korištena tehnologija 82
6.4.2 Najvažniji segmenti poslužiteljskog dijela aplikacije 84
6.4.3 Konceptualni model baze podataka 87
6.5 Opis specifičnih rješenja 88
6.5.1 Utvrđivanje identiteta korisnika 88
6.5.2 Postavke aplikacije 91
6.5.3 Plan učenja 91
6.5.4 Realizacija čitača RSS *feedova* 94
6.5.5 Komunikacija korisnika 95
7 Analiza korištenja aplikacije za podršku stvaranju osobnog okoliša za učenje 97
7.1 Osnovni slučajevi korištenja 97
7.1.1 Prijava u aplikaciju 97
7.1.2 Administracija sustava 98
7.1.3 Proces učenja 99
7.2 Primjeri korištenja aplikacije 106
7.2.1 Primjer korištenja aplikacije u samostalnom učenju 106
7.2.2 Primjer korištenja aplikacije kao pomoćnog sredstva pri pisanju teksta 107
7.3 Analiza korištenja aplikacije 108
7.3.1 Instalacija aplikacije 109
7.3.2 Rezultati testiranja 110
7.3.3 Potrebne dorade po aplikaciji 111
7.4 Usporedba aplikacije *Samouk* s drugim sustavima za stvaranje PLE-a 115
7.5 Daljnji razvoj 117
8 Zaključak 118
9 Literatura 119
10 Popis internetskih adresa 127
11 Popis oznaka 129
11.1 Popis slika 129
11.2 Popis tablica 130
12 Sažetak 131
13 Summary 132
14 Ključne riječi 133
15 Životopis 134

1 UVOD

Današnji svijet prolazi kroz velike i značajne promjene uzrokovane razvojem digitalne računalno komunikacijske tehnologije. Promjene utječu na svaki vid ljudskog života pa tako i na gospodarstvo, koje zahtjeva obrazovanu radnu snagu koja je fleksibilna i sposobna za neprekidno osposobljavanje i prilagođavanje novim potrebama gospodarstva. Da bi bili konkurentni na tržištu rada i išli ukorak s promjenama, po završetku formalnog obrazovanja ljudi se moraju neprekidno obrazovati.

1.1 Motivacija za magistarski rad

Današnje društvo je društvo znanja u kojem obrazovani ljudi čine vodeću proizvodnu snagu. Promjene u takvom društvu su vrlo brze, tako da jednom usvojena znanja i vještine vrlo brzo zastarijevaju. Znanje koje se stekne tijekom formalnog školovanja u najvećem broju slučajeva više nije dostatno za cijeli život i potrebno ga je prilagođavati promjenama u okruženju. Odrasle osobe su danas prisiljene neprekidno učiti kako bi osigurale prilagodbu svog znanja i vještina potrebama promjenjivog tržišta rada. Na takav način zaposleni osiguravaju temeljnu pretpostavku za zadržavanje zaposlenja, a nezaposleni povećavaju šansu za zapošljavanje [94][115]. Formalno obrazovanje počevši od osnovne škole, preko srednje do fakulteta daje okvir, a po njegovom završetku ljudi bi se trebali nastaviti aktivno obrazovati u specifičnom području kojem se namjeravaju posvetiti u poslu ili u životu. Takvim procesom obrazovanja upravljaju pojedinci koji samostalno planiraju karijeru i prikupljaju znanje od kolega na poslu, iz knjiga, časopisa i različitih izvora informacija dostupnih na Internetu. Internet postaje vodeći resurs za pronalaženje brojnih materijala u digitalnom obliku. Velik dio tih materijala je slobodno dostupan bez naknade i može ga koristiti svatko tko je za to zainteresiran. Odrasle osobe danas uglavnom uče informalno i procjenjuje se da je gotovo 70 do 80% ukupno stečenog znanja stečeno na takav način [20][35][65].

U formalnom obrazovanju posljednjih nekoliko desetljeća se posvećuje velika pažnja računalom podržanom učenju. Sustavi koji su razvijeni za tu namjenu služe kao pomoć, a u nekim slučajevima u potpunosti zamjenjuju nastavnika prilikom prijenosa znanja učenicima. Oni su uglavnom orijentirani instituciji odnosno nastavnicima, služe za podučavanje i pomoću njih mali broj nastavnika može učinkovito upravljati prijenosom znanja većem broju učenika. Tome je prilagođena njihova arhitektura i nastavni materijali koji su orijentirani uniformnom načinu prijenosa znanja. Takav pristup dovodi do problema koji proizlaze iz činjenice što su ljudi različiti, i nekima će način na koji se znanje prenosi odgovarati više, a drugima manje. To je dovelo do većeg zanimanja i proučavanja adaptivne hipermedije i inteligentnih tutorskih sustava koji bi trebali omogućiti prijenos znanja većem broju učenika koristeći pristup koji je prilagođeniji svakom pojedinom učeniku [9][67].

Kod samostalnog učenja, učenik uglavnom uči sam i nema nastavnika ili instruktora koji će mu pripremati nastavne materijale i voditi ga tijekom učenja. Pri takvom načinu učenja, učenik mora samostalno preuzeti njihovu ulogu, odrediti svoj cilj i metode učenja, te odabrati i pripremiti materijale. Dobre polazne osnove mu u tome pomažu, a ako učenik ima slabije

polazne osnove, tijekom procesa samostalnog učenja veliku pomoć mu mogu pružiti druge osobe koje imaju određeno znanje o području učenja. Ako su te osobe fizički udaljene, s njima je moguće jednostavno komunicirati korištenjem različitih internetskih servisa [23]. Zbog orijentacije prema podučavanju, sustavi za računalom podržano učenje kakvi se koriste u formalnom institucionalnom obrazovanju nisu pogodni za korištenje pri samostalnom učenju. Učenici prilikom samostalnog učenja traže sustav koji je jednostavan za prilagodbu i korištenje, omogućuje pristup i pohranu materijala u elektroničkom obliku, te jednostavnu komunikaciju s drugim osobama. Važna osobina je da sustav bude prilagodljiv učeniku i da su materijali unutar njega učeniku dostupni cijelog života, a ne samo za vrijeme trajanja učenja.

1.2 Pregled dostignuća na polju razvoja osobnih okoliša za učenje

Konektivizam (engl. *connectivism*) je teorija koja proučava način na koji ljudi stječu znanje u suvremenom digitalnom dobu. Po postavkama te teorije na proces učenja u velikoj mjeri utječu veze koje osoba uspostavlja s drugim osobama, a učenje predstavlja proces neprekidnog prikupljanja novih činjenica i preispitivanja postojećeg znanja [72]. Postavke konektivizma su u velikoj mjeri utjecale na način na koji je zamišljen koncept osobnog okoliša za učenje.

Današnji korisnici weba (*World Wide Web*, skraćeno WWW ili web) na raspolaganju imaju brojne tehnologije i aplikacije koje omogućavaju grupni rad u okviru neformalnih skupina. Takve aplikacije intenzivno se razvijaju posljednjih desetak godina uslijed rasta popularnosti Web 2.0 principa koji predstavlja promjenu u načinu stvaranja informacija na Internetu. U mrežnom okruženju informacije se uglavnom stvaraju decentralizirano, od strane velikog broja pojedinaca okupljenih u neformalne grupe. Web je proširio mogućnosti grupnog rada tako da članovi više ne moraju biti fizički blizu da bi mogli učinkovito komunicirati. Društvene mreže stvorene na takav način mogu se učinkovito iskoristiti u procesu samostalnog učenja.

Pojam osobnog okoliša za učenje (engl. *Personal Learning Environment*, skraćeno PLE) koristi se od 2004. godine, a razvijen je kao kritika institucionalnog prijenosa znanja putem virtualnih obrazovnih okoliša (engl. *Virtual learning Environment*, skraćeno VLE). Prema Van Harmelenu [81], osobni okoliš za učenje je sustav koji omogućuje korisniku preuzimanje potpune kontrole i upravljanje procesom vlastitog učenja što obuhvaća postavljanje osobnih ciljeva učenja, upravljanje tijekom i materijalima učenja te komunikaciju s drugim osobama. Prvi sustavi te vrste služili su kao mjesto na kojem su učenici mogli samostalno pohranjivati obrazovne materijale i komunicirati s drugim učenicima. U takvom okruženju materijali su im bili dostupni i nakon završetka procesa učenja.

Na osnovu provedenog istraživanja o sustavima za podršku učenju, Jafari [40] zaključuje da je potrebna promjena modela sustava za računalom podržano učenje koji bi trebao podržati samostalno cjeloživotno učenje i biti prilagođeniji učenicima. Istraživačka grupa sa

Sveučilišta u Helsinkiju stvorila je 1998. godine model sustava za učenje budućnosti (engl. *Future Learning Environment*). Sustav je bio temeljen na webu kao platformi, a omogućavao je učenje u suradničkom okruženju u kojem su učenici i mentori ravnopravno razmjenjivali informacije i stvarali novo znanje [78]. *Colloquia* [49] je sustav koji omogućuje grupno učenje i rad korisnika koji se okupljaju oko zajedničke obrazovne teme. Pomoću sustava oni međusobno komuniciraju, prikupljaju, pohranjuju, razmjenjuju i dijele obrazovne materijale. Prva verzija sustava razvijena je 2000. godine, a sam sustav je tehnološki je bio realiziran kao okruženje ravnopravnih sudionika (engl. *peer-to-peer*). Na Sveučilištu u Manchesteru od 2004. godine odvija se projekt razvoja aplikacije za podršku izgradnji osobnog okoliša za učenje koji bi studenti mogli koristiti uz postojeći institucionalni VLE. Projekt je do danas prošao kroz nekoliko verzija i tehnoloških rješenja. Prva verzija sustava bila je realizirana kao klijentsko-poslužiteljska aplikacija u kojoj su repozitorij materijala i programska logika bili smještena na poslužitelju na koji su se spajali klijenti u obliku stolnih (engl. *desktop*) aplikacija. S razvojem tehnologije, orijentacija sustava kreće prema servisnoj orijentaciji i korištenju web servisa [79].

Na daljnji razvoj ideje osobnog okoliša za učenje utjecala je popularnost Web 2.0 koncepta. Programska rješenja koja nastaju uglavnom se baziraju na intenzivnom korištenju društvenih mreža i Web 2.0 servisa. 2004. godine razvijena je prva verzija *Elgg* [URI10] platforme koja je omogućavala korisnicima izgradnju vlastite društvene mreže korištenjem Web 2.0 servisa. Pomoću te platforme svatko je mogao na jednostavan način stvoriti okruženje vlastite društvene mreže i organizirati ga po svojim željama i potrebama. Brojni korisnici u praksi su stvorili osobni okoliš za učenje pomoću postojećih Web 2.0 aplikacije poput blogova (*blog* je skraćenica od *weblog*), wikija, aplikacija za podršku društvenim mrežama i pomoću personaliziranih web stranica [27][52].

Scott Wilson, istraživač na Sveučilištu u Boltonu 2005. godine objavio je grafički prikaz modela virtualnog okruženja za učenje baziranog na Web 2.0 principima i alatima [87]. U središtu okruženja je učenik koji ga koristi za prikupljanje, pohranu, stvaranje i objavu materijala, te u procesu komunikacije s drugim osobama. Okruženje predstavlja portal putem kojeg ima mogućnost uvida u sve dostupne materijale bez obzira na njihov izvor. Unutar okruženja intenzivno se koriste Web 2.0 aplikacije i servisi. Od tada se većina projekata za izradu podrške stvaranju PLE-ova bazira na webu kao platformi i korištenju dostupnih Web 2.0 servisa. Kako je sve više Web 2.0 aplikacija počelo otvarati svoja programska sučelja postajao je popularan koncept hibridnih web aplikacija (engl. *mashup*) koje koriste podatke i funkcionalnost iz više različitih izvora. U posljednjih nekoliko godina, prema tom principu [69] razvijaju su prototipovi sustava za izradu osobnog okoliša za učenje, od kojih su najznačajniji mPLE (*Manchester Personal Learning Environment*) [82], MUPPLE (*Mashup Personal Learning Environment*) [85][URI23], PLME (*Personal Learning and Maturing Environment*) [4] i PLEF (*Personal Learning Environment Framework*) [13][15] [URI29].

1.3 Cilj i struktura rada

Cilj ovog rada je izrada konceptualnog modela i realizacija prototipa programske potpore stvaranju osobnog okoliša za učenje.

Magistarski rad podijeljen je u dvije cjeline. Prva cjelina opisuje teorijske postavke računalom podržanog učenja s naglaskom na osobne okoliše za učenje i Web 2.0 koncept, i uključuje prva tri poglavlja. Druga cjelina obuhvaća sljedeća tri poglavlja u kojima se opisuje model sustava te njegova praktična realizacija i evaluacija. Na kraju rada se navode mogućnosti daljnje nadogradnje i unaprjeđenja sustava.

U drugom poglavlju ovog rada opisane su teorijske postavke, povijesni razvoj i trenutno stanje sustava za računalom podržano učenje. Većina sustava za računalom podržano učenje danas je realizirana na webu čemu pridonose karakteristike globalne mreže koja omogućuje jednostavan pristup resursima i komunikaciju među korisnicima. Razvoj novih Web 2.0 aplikacija i servisa podržava takav način rada i one se mogu koristiti u procesu samostalnog učenja. U trećem poglavlju opisan je Web 2.0 koncept s naglaskom na njegove osobine i alate koji se mogu koristiti u sustavima za računalom podržano učenje. U četvrtom poglavlju opisani su pojam i karakteristike PLE-a, a koncept je uspoređen s konceptom klasičnog učenja u okviru VLE-a.

U petom poglavlju na osnovu teorijskih postavki opisanih u prethodnim poglavljima izgrađen je model sustava za stvaranje PLE-a, baziran na korištenju Web 2.0 principa. Dio funkcionalnosti modela realiziran je u sklopu ovoga rada kao praktičan prototip, a najzanimljiviji elementi implementacije opisani su u šestom poglavlju. U sedmom poglavlju su opisani primjeri praktičnog korištenja realiziranog prototipa aplikacije nazvane *Samouk*. Analizirani su slučajevi korištenja i opisane prednosti i nedostaci uočeni tijekom testnog korištenja. Na osnovu rezultata testiranja navedene su mogućnosti nadogradnje i daljnjeg razvoja izgrađenog modela i aplikacije.

Zaključno osmo poglavlje je sažetak rada u kojem se navode osnovne smjernice daljnjeg istraživanja.

2 RAČUNALOM PODRŽANO UČENJE

Na razvoj računalom podržanog učenja u dvadesetom stoljeću utjecale su tri važne teorije učenja: biheviorizam, kognitivizam i konstruktivizam. U prvom dijelu ovog poglavlja opisane su njihove teorijske postavke s naglaskom na elemente koji su utjecali na razvoj sustava za računalom podržano učenje. U drugom dijelu opisan je razvoj sustava za računalom podržano učenje, a u trećem koncept VLE-a koje je danas dominantan oblik sustava za računalom podržano učenje.

2.1 Obrazovanje i učenje

Obrazovanje obuhvaća proces, sadržaj i rezultat organiziranog ili slučajnog učenja s ciljem razvoja različitih kognitivnih sposobnosti, kao i stjecanja raznovrsnih znanja, umijeća i navika. Učenje je uži pojam od obrazovanja i obuhvaća usvajanje navika, informacija, znanja i vještina, njihovo povezivanje s prethodno usvojenim znanjem, te sposobnost smještanja znanja u kontekst pamćenja [77].

2.1.1 Podjela obrazovanja

Klasična podjela obrazovanja je trodijelna, institucionalna i bazirana na podjeli prema mjestu na kojem se obrazovanje odvija. Tu podjelu je 1973. godine predložio Philipe H. Coombs [64] i prema njoj obrazovanje možemo podijeliti na formalno, neformalno i informalno.

Formalno obrazovanje vezano je uz školovanje unutar školskog sustava, u školama, na fakultetima, veleučilištima i sveučilištima kojima je obrazovanje osnovna djelatnost. Po završetku obrazovnog procesa takve institucije polaznicima izdaju opće prihvaćenu javnu ispravu o završenom stupnju i vrsti obrazovanja (svjedodžbu ili diplomu). Formalno obrazovanje odraslih osoba najčešće obuhvaća srednjoškolsku specijalizaciju za odgovarajuće zanimanje ili nastavak visokoškolskog obrazovanja.

Neformalno obrazovanje se organizirano provodi izvan formalnog školskog sustava u ustanovama, udrugama, poduzećima i drugim organizacijama koje se bave obrazovanjem kao osnovnom ili dopunskom djelatnošću. Tijekom neformalnog obrazovanja osobe pohađaju različite programe i tečajeve koji su obično usko specijalizirani, a po završetku školovanja polaznici dobivaju potvrdu o završenom obrazovanju koja nije javna isprava. Neformalno obrazovanje često je npr. kod specijalista koje poslodavci šalju na tečajeve proizvođača strojne ili programske opreme na kojoj rade.

Informalno obrazovanje obuhvaća sve ostale načine na koje osoba stječe novo znanje. Ono može biti svjesno ili nesvjesno, a osnovna karakteristika mu je da tijekom procesa ne postoji vanjski autoritet koji upravlja tijekom i načinima učenja. Svjesno informalno učenje je samousmjereno ako osoba samostalno upravlja tijekom učenja i procesom stjecanja znanja. Takav način obuhvaća čitanje knjiga i časopisa, gledanje filmova i televizije, prikupljanje informacija na Internetu, razgovore s kolegama i suradnicima te samostalno problemsko učenje. Nesvjesno informalno učenje je posljedica *svakog* socijalnog kontakta s drugom osobom ili okolinom koji rezultira novostečenim znanjem.

Većina znanja koje osoba stekne tijekom svog života stečena je informalno. Provedene studije [20][35][65] pokazuju da je gotovo 70 do 80% ukupnog znanja odrasle osobe stečeno na informalan način i s pravom se može tvrditi da radnici više nauče za vrijeme druženja tijekom pauze za kavu nego tijekom formalne obuke [17].

2.1.2 Teorije učenja

Kroz povijest, znanstvenici iz različitih disciplina su nastojali objasniti proces stjecanja znanja. Većina teorija se slaže da je učenje proces koji dovodi do relativno trajnih promjena u ponašanju pojedinca. Proučavanjem načina na koji ljudi uče prvi su se počeli baviti psiholozi koji su u razvili niz teorija učenja od kojih su najvažnije biheviorizam, kognitivizam i konstruktivizam [64].

2.1.2.1 Biheviorizam

Biheviorizam je psihološka teorija utemeljena na eksperimentalnom istraživanju i objektivnom mjerenju ponašanja analognom onome u prirodnim znanostima. Razvoj teorije započeo je početkom dvadesetog stoljeća u Rusiji i SAD-u, a teorija je bila dominantna prvih sedamdeset godina dvadesetog stoljeća. Začetnik biheviorističke teorije je Ivan Pavlov koji je u svojim istraživanjima na životinjama otkrio proces formiranja uvjetovanog refleksa. Osnovna pretpostavka biheviorističke teorije je da je ponašanje osobe određeno nekim vanjskim faktorom na koji se može utjecati. Ako se taj faktor uspije pronaći, moći će se programirati čovjekovo ponašanje u željenom smjeru. Ako je određena radnja praćena pozitivnim efektima za onog tko je izvodi, ona će biti učvršćena u njegovom pamćenju, a vjerojatno i u ponašanju. Ako su efekti negativni, radnja će biti uklonjena iz ponašanja [62].

Bihevioristička teorija je značajno utjecala na razvoj sustava za računalom podržano učenje, a u mnogim postavkama današnjeg načina obrazovanja mogu se prepoznati bihevioristički elementi i obrasci. Npr. učenici se prije početka učenja moraju upoznati s očekivanim rezultatima učenja kako bi samostalno mogli procijeniti da li su i u kojoj mjeri postigli očekivane rezultate. Po završetku učenja svake teme potrebno je provjeriti znanje učenika kako bi se odredilo u kojoj mjeri su postignuti očekivani rezultati. Testiranje mora biti sastavni dio procesa učenja kako bi učenik neprekidno dobivao povratne informacije o svom napredovanju i po potrebi mogao vršiti korekciju. Materijali za učenje moraju biti podijeljeni u male lekcije i poredani u smislenom redoslijedu: od poznatih prema nepoznatim pojmovima, od jednostavnijih prema složenijima i od teorijskog izlaganja prema praktičnoj primjeni [2].

2.1.2.2 Kognitivizam

Kognitivizam je psihološka teorija usmjerena na otkrivanje psihičkih procesa koji se nalaze u osnovi ljudskog ponašanja. Teorija proučava unutarnje procese za koje se pretpostavlja da utječu na oblikovanje ponašanja (pamćenje, pozornost, percepciju, predstavljanje znanja, mišljenja i sl.). Osnovna razlika između biheviorizma i kognitivizam je što kognitivisti smatraju da se osoba ne može promatrati kao crna kutija samo na osnovu ulaza i izlaza, već se mora zaviriti u tu kutiju kako bi se pokušali shvatiti procesi koji u njoj djeluju. Osoba reagira

na odgovarajuće podražaje, ali način na koji će čovjek reagirati na podražaj jako ovisi o čovjekovoj unutarnjoj strukturi: razmišljanju, stavovima i osjećajima.

Npr. student je pao na ispitu i dobio negativnu ocjenu. Dobivena ocjena predstavlja podražaj. Nastavnik očekuje da će student uslijed tog podražaja više učiti kako bi sljedeći put položio ispit. Ako student ima pozitivan stav prema učenju, ispitu ili nastavniku, on će reagirati na očekivani način i provesti više vremena u učenju kako bi sljedeći put položio ispit. Ako s druge strane student ima negativan stav tipa „nije me briga“, „škola nije važna“ ili „to nije u redu“ on se neće ponašati na očekivani način već će odustati, ili će do željenog cilja (pozitivne ocjene) pokušati doći na neki drugi način koji ne uključuje učenje.

U ovom primjeru očekivana reakcija studenta ovisi o njegovom unutarnjem stanju, razmišljanjima i stavovima, pa nije uvijek onakva kakvu nastavnik očekuje.

Tipične metode poučavanja prema kognitivističkim načelima su korištenje primjera i modela za usvajanje i povezivanje pojmova, vježbe kategorizacije i usporedbe, izrada dijagrama i shema te oslanjanje na ranije naučeno u stjecanju novih znanja. Učenje uz pomoć računala omogućuje predstavljanje informacija na različite načine, tako da učenici prilikom njihovog usvajanja moraju koristiti više različitih osjetila (npr. privlačne boje, zvuk i animacija). Informacije se na ekranu trebaju prikazati u logičnom poretku (npr. s lijeva na desno), a kako bi ih učenik lako uočio, najvažnije informacije moraju biti naglašene i nalaziti se u središtu. Sadržaje treba prezentirati u malim, međusobno povezanim cjelinama, a potrebno je naglašavati veze među dijelovima za što su pogodni vizualni prikazi struktura poput usmjerenih grafova. Osnovna strategija učenja trebala bi biti uklapanje novih informacija u postojeću kognitivnu strukturu koje je smještena u učenikovoj dugotrajnoj memoriji. To je moguće postići ako je učenik prethodno upoznat s ciljem i očekivanim rezultatima učenja. Osvježavanje postojećeg znanja može se provesti ulaznim testovima, a nove informacije bi trebalo predstavljati u malim blokovima s maksimalno pet do devet novih pojmova unutar jedne lekcije, a za uspjeh učenja važno je da su učenici motivirani za učenje [2].

2.1.2.3 Konstruktivizam

Konstruktivizam je teorija učenja prema kojoj učenici samostalno kreiraju znanje na temelju vlastitog iskustva, pa je zbog toga način stjecanja znanja jedinstven kod svakog pojedinca. Kada učenik sazna novu činjenicu, on ju obrađuje na osnovu prethodnog znanja i vlastitih kognitivnih struktura: stavova, misli i osjećaja [39]. Učenje je rekurzivan postupak kojim se novo znanje nadograđuje na osnovu prethodno poznatih podataka i činjenica.

Na prvi pogled se ne može uočiti velika razlika između prethodno opisane kognitivističke teorije i konstruktivizma, međutim radi se o velikoj promjeni jer je prvi put uloga subjekta nastave promijenjena. Nastavnik je izmješten iz središta, a učenik od pasivnog subjekta nastave postaje aktivni sudionik i njezin kreator. On zajedno s učiteljem i drugim učenicima samostalno kreira novo znanje, oblikuje i preoblikuje dostupne materijale za učenje i sam proces učenja. Uloga nastavnika je također promijenjena i on više nije isključivi prenositelj znanja već se nalazi u ulozi pomagača koji učenike na početku uvodi u problematiku, a nakon toga ih potiče i usmjeravaju da samostalno otkrivaju načela i zakone u sadržajima koje uče.

Pri tome im može pomoći tako da informacije koje trebaju naučiti prethodno transformira u oblik koji je prilagođen razini prethodno usvojenog znanja učenika.

Na učenike, naročito odrasle koji uče uz rad, motivirajuće djeluje povezivanje problema koje rješavaju za vrijeme učenja s problemima u stvarnom svijetu. Učenje na takav način događa se svakodnevno npr. kod programera u računalnoj industriji. Svaki programer je na početku svoje karijere naučio osnove programiranja i neki programski jezik. Znanje je stekao tijekom formalnog obrazovanja, iako ni samouki programeri nisu rijetkost. Takva osoba se zapošljava u nekom poduzeću i dobiva radne zadatke. Vrlo rijetko će se dogoditi idealna situacija da se njegova znanja u potpunosti poklapaju s potrebama razvoja. Češći je slučaj da nakon dolaska na novo radno mjesto mora naučiti novi programski jezik ili tehnologiju. Kod nas je u praksi slanje programera na formalno školovanje u takvim slučajevima dosta rijetko. Mnogo češće programer će dobiti literaturu, ili će mu se jednostavno reći „snađi se“. U opisanom slučaju programer ima početno znanje o nekom programskom jeziku i zna da su svi programski jezici u svojoj osnovi relativno slični. Polazne informacije će pomalo nadograđivati, uspoređivati će sintaksu i način programiranja s postojećim i pomalo stjecati znanje o novoj tehnologiji. Ako pri tome ima pomoć nekog iskusnog kolege koji poznaje tu tehnologiju, stjecanje novog znanja biti će brže i jednostavnije. U ovom primjeru vidljive su tri važne odrednice konstruktivizma:

- Osoba je motivirana za učenje (iz osobnih motiva ili pod vanjskim utjecajem);
- Osoba koja uči ima odgovarajuće predznanje o problematici o kojoj uči, a učenje predstavlja nadogradnju prethodnog znanja. Što je količina prethodnog znanja veća i sličnija problemu, to će nadogradnja biti jednostavnija, brža i učinkovitija;
- Učenje je aktivno. Osoba zna razlog zbog kojeg nešto želi naučiti i poznato joj je (barem okvirno) okruženje u kojem će stečeno znanje primjenjivati. Na takav način se može fokusirati na važne segmente. Npr. ako programer zna da će se u praksi baviti izradom poslovnih aplikacija, više će se fokusirati na proučavanje aspekata vezanih uz rad s bazom podataka nego npr. računalnom grafikom.

2.2 Elektroničko učenje

2.2.1 Definicije pojma

Za pojam elektroničkog učenja (engl. *E-learning*) postoji više različitih definicija koje naglasak stavljaju na proces učenja ili na primjenu tehnologije u tim procesima. Uobičajena definicija elektroničkog učenja s tehnološkog stajališta je da je to „bilo koji oblik učenja, poučavanja ili obrazovanja koji je potpomognut uporabom računalnih tehnologija, a posebno računalnih mreža temeljenih na internetskim tehnologijama“ [26]. Drugi pogled na elektroničko učenje je da je to „pohrana informacija u računalni sustav čime one postaju dostupne učenicima kada su im potrebne. Pohrana informacija vrši se na tri načina: izravnim unosom informacija u sustav od strane korisnika, bilježenjem informacija koje nastaju uslijed akcija učenika tijekom učenja, te pohranom informacija koje stvara sam sustav na osnovu

promatranja akcija korisnika“ [53]. Ova definicija elektroničko učenje promatra sa stajališta izvora informacija. Prilikom formalnog ili neformalnog učenja kada učenjem upravlja vanjski autoritet (nastavnik ili instruktor) za učenika su na početku najvažnije informacije koje je u sustav unijela ta osoba (npr. nastavni materijali ili primjeri starih ispitnih zadataka). Kako učenje odmiče i učenici imaju priliku kontaktirati s drugim učenicima, sve važniji postaju materijali koje stvaraju oni sami. Ta vrsta materijala je dominantna u okruženjima u kojima procesom učenja upravlja sam učenik. Informacije koje stvara sam sustav mogu se koristiti pri izradi sustava koji su u većoj mjeri prilagođeni korisnicima [9].

2.2.2 Podjela elektroničkog učenja

Elektroničko učenje može se podijeliti prema tome koliko se razlikuje od klasičnog učenja u učionici. Ako se elektroničko učenje koristi kao jedan od ravnopravnih elemenata obrazovnog procesa, naziva se hibridno učenje (engl. *blended learning* ili *hybrid learning*). Takav način korištenja računalne tehnologije u nastavi danas je vrlo popularan u okviru formalnog i neformalnog obrazovanja. Čisto elektroničko učenje (engl. *pure e-learning*) je druga kategorija kod koje se proces odvija u potpunosti van učionice. Učenici i nastavnici nemaju izravan kontakt licem u lice, a njihova međusobna komunikacija se odvija korištenjem informacijsko komunikacijskih tehnologija. Čisto elektroničko učenje može biti sinkrono i asinkrono. Sinkrono elektroničko učenje u potpunosti prenosi okruženje i način rada u učionici. Svi sudionici tijekom nastave moraju biti istovremeno prisutni u virtualnoj učionici. Osnovna prednost takvog načina je u tome što učenici i nastavnici ne moraju biti fizički na istom mjestu već mogu biti dislocirani. Prilikom korištenja asinkronog elektroničkog učenja učenicima su materijali neprekidno dostupni i oni ih u takvom okruženju mogu koristiti u bilo koje doba dana [26].

2.2.3 Karakteristike elektroničkog učenja

Većina sustava za učenje pomoću računala danas se nalazi na webu, a za pristup tim resursima učenici trebaju imati računalo i vezu na Internet. Brzina veze određuje vrstu materijala koje će učenici moći dominantno koristiti (npr. pristup multimedijalnim materijalima poput zvučnih ili video zapisa zahtijeva pristup širokopojasnom Internetu). Učenje u takvom okruženju nekada je bilo problematično zbog slabije računalne i komunikacijske opremljenosti, no to danas uglavnom nije problem. Prema statističkim podacima za 2008. godinu, u Republici Hrvatskoj (skraćeno RH) 53% kućanstava, odnosno 98% poslovnih subjekata posjeduje osobno računalo. Pristup Internetu ima 45% svih kućanstava u RH, a pri tome je dominantan širokopojasni pristup Internetu koji obuhvaća 60% svih internetskih priključaka [114]. Na nivou Europe situacija je još bolja i širokopojasni pristup Internetu ima gotovo 50% svih kućanstava [106].

Elektroničko učenje ima određene prednosti u odnosu na učenje u učionici. Učenik je neovisan o mjestu na kojem uči i fizički može boraviti na bilo kojoj lokaciji na kojoj ima pristup Internetu. Putem sustava može pristupati obrazovnim sadržajima i komunicirati s nastavnikom i drugim učenicima. Na takav način ostvaruju se financijske i vremenske uštede

jer ne mora putovati na lokaciju na kojoj se nastava odvija i boraviti na njoj određeno vrijeme. Kod asinkronog modela učenja učenik samostalno određuje vrijeme i tempo učenja. Fleksibilno vrijeme učenja posebno odgovara učenicima koji se obrazuju uz rad i koji uglavnom uče u svoje slobodno vrijeme. Suvremeni sustavi korištenjem inteligentnih tehnologija unaprjeđuju nastavne sadržaje kako bi bili što prilagođeniji pojedinom učeniku.

Unutar okruženja za elektroničko učenje nastavnici mogu kontinuirano pratiti napredovanje učenika tijekom učenja. Sustav obično bilježi sve aktivnosti korisnika i iz tih bilješki se izrađuju izvješća za administratore, nastavnike i instrukcijske dizajnere (engl. *instructional designers*). Nastavnici te informacije mogu koristiti kao pomoć pri usmjeravanju učenika tijekom procesa učenja, a instrukcijskim dizajnerima to može biti korisna povratna informaciju o učestalosti i načinu korištenja, te posredno o kvaliteti kreiranih materijala.

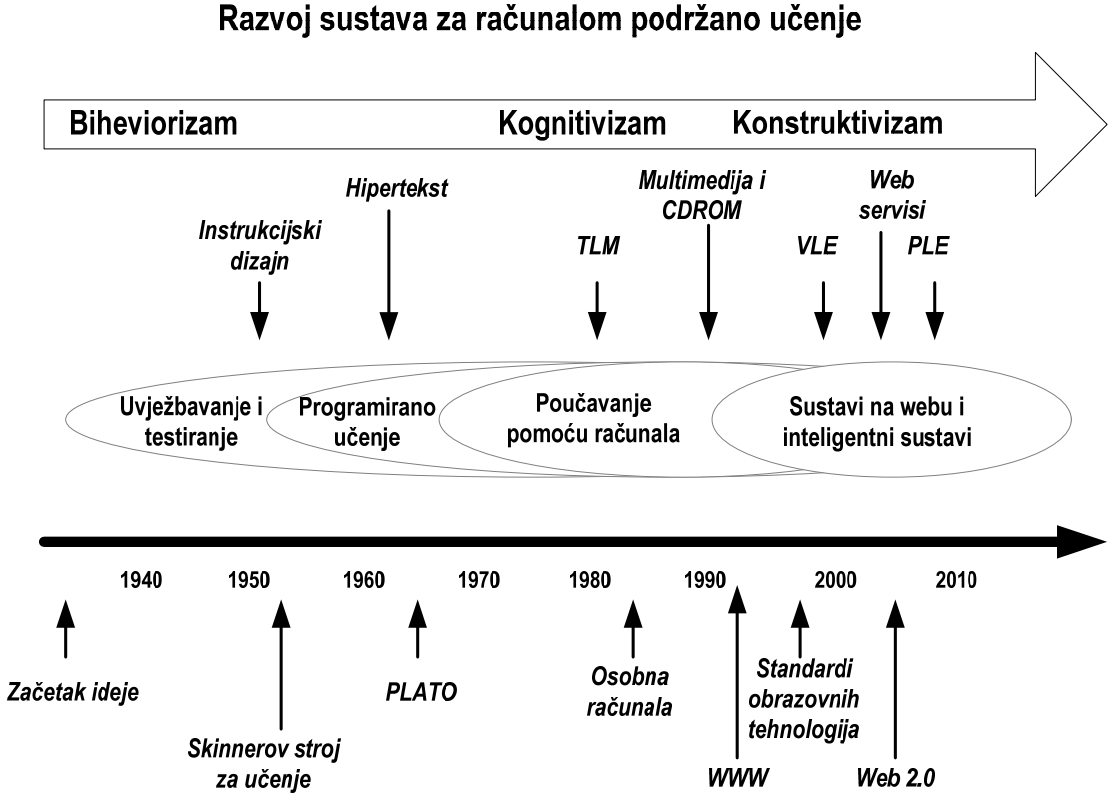
Nedostaci elektroničkog učenja su vezani uz karakteristike učenja i samih učenika. Kod čistog elektroničkog učenje nema međusobnog kontakta licem u lice između učenika niti između učenika i nastavnika. Sinkrone metode komunikacije tu mogu pomoći, ali ne mogu u potpunosti zamijeniti osobni kontakt. Pošto učenik samostalno određuje vrijeme i tempo učenja, mogu se pojaviti problemi motivacije i uslijed toga odustajanje od pohađanja nastave. U klasičnom obrazovnom modelu učenik je kroz proces učenja vođen od strane vanjskog autoriteta koji mu pomaže i upravlja procesom učenja. U okruženju učionice uočen je poticajan utjecaj drugih učenika i nastavnika, dok je kod učenja u virtualnoj učionici taj utjecaj manji. Kod samostalnog učenja vrlo su važni disciplina i posvećenost samog učenika, i ako on ima viziju praktične primjene znanja koje usvaja, to može biti motivirajući faktor.

Tijekom učenja uz pomoć računala, na učenika znatno utječu ergonomske karakteristike okruženja. Dulje korištenje materijala u elektroničkom obliku na zaslonima loše kvalitete može biti zamorno. Instalacija, korištenje i održavanje sustava za elektroničko učenje je zahtjevno i skupo što unosi dodatne troškove u obrazovni sustav. Stvaranje kvalitetnih materijala za elektroničko učenje je složen, skup i dugotrajan proces koji zahtijeva suradnju različitih stručnjaka. Posljedica toga je da se kvalitetni obrazovni materijali u elektroničkom obliku rijetko nalaze.

2.2.4 Razvoj sustava za računalom podržano učenje.

Na razvoj današnjih sustava za računalom podržano učenje utjecale su sve prethodno opisane teorije učenja. Prema postavkama biheviorističke teorije oblikovani su sustavi za obuku kojima je osnovna namjena prijenos činjenica i uvježbavanje određenih akcija. Kao posljedica kognitivističkog pristupa u sustavima za učenje se počela obraćati veća pažnja učenju principa i procesa te uklapanju novih saznanja u postojeća. Konstruktivistička načela su utjecala na razvoj sustava u kojima učenici imaju veću autonomiju. Omogućena je prilagodba materijala, procesa učenja i suradnički rad, te uklapanje vlastitih materijala u sustav.

Na slici 2.1 prikazani su najvažniji događaji i tehnologije u razvoju sustava za računalom podržano učenje.



Slika 2.1 Povijesni razvoj sustava za računalom podržano učenje

Sustavi koji služe za učinkovitu isporuku nastavnih sadržaja učenicima u okviru institucija zajedničkim imenom se nazivaju sustavi za računalom podržano poučavanje (engl. *Computer Based Instruction*, skraćeno CBI). Sustavi tog tipa danas su vrlo popularni i koriste se kao podrška procesima formalnog obrazovanja. Ovisno o okruženju u kojem se koriste, nose različita imena koja se u globalu odnose na isti koncept. U gospodarstvu je usvojen termin računalom podržana obuka (engl. *Computer-Based Training*, skraćeno CBT), dok su u akademskim sredinama u upotrebi termini računalom podržano obrazovanje (engl. *Computer-Based Education*, skraćeno CBE) ili poučavanje pomoću računala (engl. *Computer-Assisted Instruction*, skraćeno CAI). U obrazovnom dijelu oni podržavaju različite aktivnosti od kojih su najvažnije vježbanje i uvježbavanje (engl. *drill-and-practice*), stručni tečajevi (engl. *tutorial*), provjere znanja, simulacije, igre, grupni rad i zadaci problemskog tipa. Kako bi se poboljšao prijenos znanja izrađuju se sustavi koji koriste napredne metode i tehnike iz područja statistike i umjetne inteligencije. U sustave te vrste spadaju prilagodljivi hipermedijski sustavi (engl. *Adaptive Hypermedia System*, skraćeno AHS) i inteligentni tutorski sustav (engl. *Intelligent Tutoring System*, skraćeno ITS) [67].

2.2.4.1 Nastanak ideje računalom podržanog učenja

Ideju sustava za računalom podržano učenje razmatrao je američki psiholog Edward Thorndike koji je 1912. godine zamislio stroj koji će omogućiti samostalno učenje i provjeru

usvojenog znanja odgovorima na pitanja s višestrukim izborom. Učenik bi nakon što odgovori na sva zadana pitanje dobio povratnu informaciju i imao uvid u točne odgovore. Sidney L. Presser je 1925. godine razradio Thorndikeovu ideju i izradio stroj koji je koristio za ispitivanje studenata psihologije. Student je na postavljeno pitanje odgovarao pritiskom na jednu od 4 tipke na stroju. Stroj bi zabilježio njegov odgovor i prešao na sljedeće pitanje. Nakon što bi student odgovorio na sva pitanja, stroj bi ispisao odgovore i dao informaciju o ostvarenom uspjehu. Unatoč početnom zanimanju, stroj nije ušao u širu primjenu [2].

2.2.4.2 Programirano učenje

Razvoj prvih sustava za učenje započet je na inicijativu Američkog ministarstva obrane tijekom Drugog svjetskog rata. Sredinom pedesetih godina 20. stoljeća vodeći biheviorist tog vremena B. F. Skinner osmislio je postavke stroja za učenje koji je pomagao učenicima pri samostalnom učenju. Učenik bi dobio niz zadataka koje je morao riješiti zadanim redoslijedom, a na kraju bi dobio povratnu informaciju o uspjehu. Osnovne postavke dizajna tog stroja nalaze se u sustavima za programirano učenje (engl. *programmed learning*) [16], koji su utjecali na razvoj koncepata CAI i CBT. Kod programiranog učenja polaznicima se određenim redoslijedom prezentiraju sadržaji, postavljaju pitanja i zadaci. Takav način rada potiče njihovu mentalnu aktivnost, a kao nagrada i motivacija služi uvid u točne odgovore.

Skinner [73] je odredio osnovne postavke računalom podržanog učenja i instrukcijskog dizajna koje su se zadržale do danas. Provjera stečenog znanja treba se obavljati nizom pitanja na koje učenici trebaju odgovarati (princip poticaj-reakcija). Pitanja trebaju biti otvorenog tipa, a ne na principu biranja između alternativnih odgovora. Lekcije trebaju biti kratke i završiti pitanjima na koje učenik treba odgovoriti. Prijelaz s lekcije na lekciju učenika postepeno vodi kroz gradivo. Tek nakon što usvoji znanje iz jednog područja i točno odgovori na sva postavljena pitanja, učenik može prijeći na slijedeću lekciju. Učenik mora dobiti povratnu informaciju o tome da li je odgovor točan ili ne neposredno nakon što odgovori na pitanje. Ako odgovor nije točan, učenik treba dobiti uputu na koji način može poboljšati znanje kako bi slijedeći put točno odgovorio. Uspješni odgovori na pitanja i završetak tečaja trebaju biti popraćeni odgovarajućom nagradom poput pohvale ili dobre ocjene.

Norman Crowder se nije slagao sa Skinnerovom postavkom da svaki učenik mora proći istim redoslijedom kroz sve sadržaje pa je 1959. godine unaprijedio koncept i izradio sustav koji je omogućavao grananje unutar lekcija ovisno o učenikovom izboru [16]. Na takav način učenik je mogao preskočiti poznate dijelove i time unaprijediti svoj proces učenja.

2.2.4.3 Hipertekst

Psiholog Ted H. Nelson 1960. godine započeo je projekt *Xanadu* s ciljem izrade sustava koji će omogućiti pohranjivanje, označavanje, pretraživanje i dohvat dokumenata. Sustav je zamišljen kao biblioteka umreženih dokumenata u kojoj će se iz jednog dokumenta jednostavno moći pristupiti bilo kojem drugom vezanom dokumentu. Inspiraciju za takav sustav dobio je iz ideje MEMEX stroja koji je 1945. godine opisao Vannevar Bush [10]. Bush je zamislio okruženje koje će omogućiti jednostavnu pohranu, dohvat, pretraživanje i pristup

svim dokumentima jednog korisnika. Prilikom korištenja takvog sustava korisnik bi trebao imati potpunu slobodu u odlučivanju o verziji dokumenta kojoj će pristupiti te o putu kojim će se nastaviti kretati između dokumenata. U sklopu projekta Nelson je osmislio koncept uzajamnog povezivanja podataka tako da je jednostavnom radnjom moguće doći od jednog do drugog podatka [59]. Taj koncept je kasnije nazvan hipertekst (engl. *hypertext*) i postao je popularan krajem dvadesetog stoljeća, najviše zahvaljujući razvoju weba koji je 1989. godine inspiriran tom idejom osmislio Tim Berners-Lee [6].

2.2.4.4 Poučavanje pomoću računala

Pomak od programiranog učenja prema modelu poučavanja pomoću računala utjecao je na razvoj sustava za računalom podržano učenje. Prvi sustavi te vrste po svojoj funkcionalnosti bili su vrlo slični današnjima. PLATO (*Programmed Logic for Automated Teaching Operations*) je prvi opći CBI sustav. Razvijen je na Sveučilištu u Illinoisu početkom šezdesetih godina 20. stoljeća i koristio se za poučavanje studenata tog Sveučilišta i okolnih obrazovnih institucija. Obrazovni sadržaji bili su smješteni na središnjem poslužitelju, a korisnici su im pristupali pomoću terminala. Sustav je bio dosta popularan, niz godina je bio u komercijalnoj uporabi diljem svijeta, a posljednja instalacija je prestala s radom 2006. godine. U osnovi sustava nalazilo se računalo koje je služilo kao izvor informacija učenicima, dok se provjera usvojenog znanja vršila standardiziranim testovima s višestrukim izborom. S razvojem tehnologije i komercijalizacijom sustava, sustav je unaprjeđivan te je u sustav ugrađena mogućnost korištenja multimedijalnih sadržaja. Kasnije verzije sustava PLATO omogućavale su isporuku sadržaja putem Interneta, a unutar njega su izrađene aplikacije koje su svoju popularnost stekle naknadno. Aplikacija *PLATO Notes* bila je prvi mrežni forum, a *Personal Notes* aplikacija je omogućavala asinkronu komunikaciju na principu e-mail poruka. *Talkomatic* je preteča danas popularnih aplikacija za komunikaciju čavrljanjem (engl. *chat*), a *Term-Talk instant messaging* aplikacija [121].

Learning Management System, skraćeno LMS proizveden 1980. godine bio je prvi sustav koji je obuhvaćao većinu funkcionalnosti današnjih virtualnih obrazovnih okruženja [120]. Pristupalo mu se modemskom vezom korištenjem terminalske emulacije. Sustav je omogućavao izradu i pohranu obrazovnih materijala oblikovanih kao modula koji su se mogli višestruko koristiti, a studenti i nastavnici su unutar sustava mogli međusobno komunicirati. Sustav je promijenio ime i danas je poznat pod imenom TLM (*The Learning Manager*).

Tijekom godina sustavi za računalom podržano učenje prošli su migraciju od velikih računala preko mini računala do radnih stanica. Izrada obrazovnih materijala zahtijevala je znatne resurse istraživača, razvojnih inženjera i dizajnera obrazovnih materijala te je bila vrlo spora i skupa. Posljedica toga bila je visoka cijena korištenja takvih sustava pa se oni nisu koristili u širem okruženju.

2.2.4.5 Primjena osobnih računala

Osamdesetih godina 20. stoljeća osobna računa su postala tehnološki i cjenovno dostupna većem broju korisnika. Tehnologija osobnih računala omogućila je izradu jeftinijih sustava za

računalom podržano učenje koji su se počeli masovnije koristiti. U to vrijeme zrakoplovna industrija je za obuku zaposlenika počela koristiti osobna računala. Uvođenje takvog načina obuke posljedica je napretka u tehnologiji proizvodnje zrakoplova. Tehnološka rješenja i modeli zrakoplova različitih proizvođača znatno su se razlikovali i brzo mijenjali, pa su se korisnici (piloti, inženjeri i mehaničari) morali neprekidno obučavati za rad s novim, ponekad vrlo različitim sustavima. Promjene su bile česte i znatne pa je trebalo osmisliti način obuke korisnika po prihvatljivim uvjetima.

Rješenje je nađeno u računalnoj tehnologiji. Proizvođači zrakoplova su uz svoje proizvode počeli isporučivati računalne aplikacije i nastavne sadržaje koji su omogućavali obuku korisnika za rad s njihovim modelima zrakoplova. Takav način obuke postigao je velik uspjeh. Povećanje mogućnosti računala, multimedija i pronalazak CD-ova pomogli su u stvaranju i isporuci sve boljih obrazovnih materijala. Problem koji se javio i stvarao probleme u praksi bilo je nepostojanje standarda obrazovnih materijala [26]. Uzrok je bio taj što je većina dostupnih sustava bila zatvorena i u vlasništvu proizvođača koji su koristili vlastite standarde obrazovnih materijala. Kada bi kupac želio koristiti obrazovne materijale drugog proizvođača, obično je uz njih morao nabaviti i novu aplikaciju, što je bilo vrlo skupo.

2.2.4.6 Razvoj standarda za računalom podržano učenje

Kako bi riješili problem nepostojanja standarda, grupa zainteresiranih partnera: kompanije iz područja zrakoplovne industrije, vladine agencije, obrazovne institucije i proizvođači obrazovnih sadržaja osnovali su 1988. godine odbor (*The Aviation Industry CBT Committee*, skraćeno AICC) koji je trebao proučiti problem i donijeti smjernice za njegovo rješenje. AICC je 1993. godine objavio preporuku kojom se omogućuje razmjena podataka između različitih sustava za računalom podržano učenje. Zahtjevi iz te preporuke obuhvaćali su opis potrebnih funkcionalnosti takvog sustava, a ugrađeni su u većinu današnjih sustava za računalom podržano učenje. Korištenjem preporuke od strane raznih proizvođača sustava omogućavala se razmjena podataka među sustavima, i dijelom korištenje obrazovnih materijala različitih proizvođača [26]. AICC preporuka je prvi standard koji su koristili proizvođači sustava za računalom podržano učenje.

Drugi ključni događaj u razvoju standarda sustava za računalom podržano učenje posljedica je Zaljevskog rata 1991. godine. Po završetku rata Ministarstvo obrane SAD-a iniciralo je projekt poboljšanja obuke pripadnika Nacionalne garde korištenjem računalne tehnologije. U sklopu projekta osnovana je organizacija (*Advanced Distributed Learning Initiative*, skraćeno ADL) koja je napravila prijedlog standardizacije oblika obrazovnih sadržaja u sustavima za učenje prema referentnom modelu djeljivih objekata (engl. *Sharable Content Object Referent Model*, skraćeno SCORM). SCORM preporuka predlaže da osnovni objekt učenja (engl. *Sharable Content Object*, skraćeno SCO) bude modularno izgrađen, upotrebljiv u različitim okruženjima, neovisan o tehnologiji i dostupan s različitih lokacija putem mreže. Korištenje te preporuke u sustavima za računalom podržano učenje omogućuje stvaranje obrazovnih sadržaja koji se mogu razmjenjivati i koristiti unutar različitih okruženja. Rezultat upotrebe trebao bi biti jednostavniji razvoj kvalitetnijih obrazovnih materijala. Preporuka se pokazala

korisnom i primjenjivom i u civilnom sektoru pa je ADL 1999. godine javno objavio prvu verziju SCORM preporuke. U razvoj preporuke su osim ADL-a uključene i druge organizacije koje sudjeluju u istraživanjima iz područja standardizacije na polju računalom podržanog učenja. Jedna od tih organizacija je IEEE s odborom za standardizaciju obrazovnih tehnologija (*IEEE Learning Technology Standards Committee*, skraćeno IEEE LTSC) [16]. Dio rezultata njihovog rada uključen je u SCORM preporuku, a drugi dio je objavljen kao poseban standard za opis meta podataka obrazovnih objekata (engl. *Learning Object Metadata*, skraćeno LOM) [100]. Razvoj SCORM preporuke se nastavlja, a trenutno posljednja objavljena verzija, SCORM 1.3 uvodi mogućnost adaptivne organizacije obrazovnih sadržaja pomoću skupa pravila kojima se definira redoslijed kojim učenik mora proći kroz nastavne sadržaje [97].

Organizacija IMS *Global Learning Consortium* je 1997. godine objavila prvu verziju IMS preporuke za korištenje obrazovnih tehnologija u distribuiranom okruženju. Preporuka promovira korištenje obrazovnih tehnologija koje se mogu jednostavno povezati s drugima. Novije verzije preporuke uključuju podršku Web 2.0 aplikacija poput blogova, wikija, internetskih foruma i stvaranju hibridnih web aplikacija [104].

2.2.4.7 Web kao platforma za učenje

Sredinom devedesetih godina 20. stoljeća s razvojem web tehnologija sustavi za računalom podržano učenje počinju se masovno koristiti na webu. Karakteristike okruženja dovele su do stvaranja brojnih sustava koji koriste njegove dobre osobine poput jednostavne suradnje i komunikacije korisnika. Sa stajališta mogućnosti isporuke multimedijalnih sadržaja situacija je bila loša zbog malih brzina kojima su se korisnici spajali na Internet, pa su se sustavi bazirani na multimedijalnim sadržajima uglavnom zadržali u okruženju osobnog računala. Mogućnosti weba su iskoristile obrazovne institucije koje su putem te platforme isporučivale obrazovne sadržaje velikom broju učenika i u tom okruženju su se razvili prvi VLE-ovi.

2.2.4.8 Napredne tehnologije u sustavima za računalom podržano učenje

Korištenje osobnih računala, standardizacija obrazovnih materijala, hipermedija i korištenje weba kao platforme za isporuku obrazovnih materijala unaprijedili su izvođenje računalom podržanog učenja i utjecali su na razvoj naprednih tehnologija za učenje. Kritike na sustave za računalom podržano učenje uglavnom su se odnosile na njihovu pedagošku prirodu i slabe mogućnosti prilagodbe osobnim zahtjevima učenika. Kao odgovor na te kritike, u posljednjih desetak godina u razvoju sustava za računalom podržano učenje intenzivnije su se počela primjenjivati konstruktivistička načela koja u središte procesa učenja stavljaju učenika. Početni pomaci u tom smjeru uglavnom su bili kozmetičke prirode poput mogućnosti da učenik promijeni postavke okruženja (npr. vrstu, veličini i boju slova ili smjesti određeni sadržaj na željeni dio stranice). Takvi pomaci nisu zadovoljili učenike jer su oni očekivali mogućnost samostalnog podešavanja postavki procesa učenja poput tempa učenja, vrste materijala ili puta kojim će prolaziti kroz lekcije. Sve veći naglasak se počeo stavljati na aktivno učenje i veću autonomiju učenika jer su rezultati istraživanja pokazali da učenici koji

koriste materijale prilagođene vlastitom stilu učenja postižu značajno bolje rezultate [93]. Dvije skupine tehnologija koje dominiraju u takvom razvoju su AHS i ITS.

Korištenje hipermedije u obrazovnim materijalima pomoglo je rješenju problema slijednog predstavljanja materijala. Međutim s povećanjem količine podataka i veza među njima, učenik se lako može izgubiti među dostupnim resursima jer je preplavljen informacijama i ne može jednostavno odrediti koja od dostupnih informacija je važna za njega, a koja nije. AHS sustavi nastoje riješiti problem izgubljenosti korisnika u hiperprostoru tako što primjenom metoda umjetne inteligencije prilagođavaju navigaciju unutar okruženja učenikovim potrebama. Osnovne komponente AHS-a su model domene, model korisnika i model prilagodljivosti. Model domene opisuje strukturu obrazovnog sadržaja, dok model korisnika sadrži osobine korisnika na osnovu kojih komponente modela prilagodljivosti vrše prilagođavanje obrazovnih sadržaja konkretnom korisniku. Hipermedijska komponenta korisničkog sučelja zadužena je za predstavljanje sadržaja i navigaciju korisnika. Korisničko sučelje prikazuje sadržaje i nudi različite alate za navigaciju u virtualnom okruženju [36].

ITS-ovi su nova generacija računalnih sustava namijenjena potpori i poboljšanju procesa učenja i poučavanja koja vodi računa o individualnosti učenika. Po svojoj funkcionalnosti oni predstavljaju napredan CAI sustava podržan metodama i tehnikama umjetne inteligencije. U takvim okruženjima proces učenja i poučavanja prilagođava se individualnim značajkama učenika tako što se prilikom modeliranja sadržaja za učenje vodi računa o osobinama učenika [67]. Za učenika okruženje je vrlo slično klasičnoj nastavi „jedan na jedan“ u kojoj jednog učenika obučava jedan nastavnik. Kod primjene sustava te vrste u praksi se javljaju dva problema. Prvi se odnosi na sadržaj učenja koji se pri dizajnu sustava mora prilagoditi prosječnom učeniku tako da potpuna personalizacija sadržaja nije moguća. Svi učenici će uvijek učiti koristeći isti sadržaj na sebi prilagođeniji način. Drugi se odnosi na činjenicu što učenik pri takvom načinu učenja nije aktivan sudionik već pasivan primatelj znanja u onom obliku u kojem je ekspertni sustav to predvidio. Odluke koje će sustav donijeti, a koje utječu na način predstavljanja znanja nikada neće biti savršene jer sustav nije u mogućnosti uzeti u obzir sve varijable koje utječu na učenje.

2.3 Virtualno obrazovno okruženje

Svaki od tipova obrazovanja opisanih na početku ovog poglavlja ima specifične zahtjeve prema okruženju za računalom podržano učenje. Formalno i neformalno obrazovanje su institucionalizirani i funkcioniraju tako da manji broj nastavnika ili instruktora prenosi znanje većem broju učenika. U većini institucija taj prijenos se odvija u okruženju učionice u kojem učenici i nastavnici uspostavljaju izravan kontakt. Računalno okruženje služi kao pomoć nastavnicima u upravljanju procesom poučavanja. Sa stajališta učenika takvo okruženje je korisno zato što im omogućuje pristup nastavnim materijalima koje koriste u procesu učenja kao i kontakte s nastavnikom i drugim učenicima van termina nastave u učionici. Nastavnicima je važno što na jednom mjestu imaju zabilježenu informaciju o kolegijima i polaznicima nastave, te što mogu jednostavno distribuirati nastavne materijale i obavijesti svim učenicima. Kod institucija koje nastavu održavaju u potpunosti van učionice, takvo

okruženje bi trebalo sadržavati i administrativni dio u kojem će moći zabilježiti podatke o polaznicima i pratiti njihovo napredovanje tijekom nastave. Takva okruženja se obično nazivaju LMS-ovi ili VLE-ovi. Ta dva naziva su sinonimi, a izraz LMS se češće koristi u Sjevernoj Americi, a izraz VLE u Europi [84]. U nastavku ovoga rada, takvo okruženje nazivati će se VLE.

2.3.1 Definicija virtualnog obrazovnog okruženja

Ne postoji opće prihvaćena definicija pojma VLE. Često se zbog riječi virtualno, takvo okruženje definira kao „svako okruženje kod kojeg se učenje ne odvija u stvarnoj učionici". Pojam se općenito odnosi na „bilo koji računalni sustav unutar kojeg se na sistematičan način vrši prijenos znanja korištenjem različitih metoda, tehnika i alata za isporuku obrazovnog sadržaja" [84]. Navedena definicija je vrlo općenita i unutar sebe obuhvaća različite skupine računalnih sustava za podršku učenju koji imaju sličnu namjenu i način funkcioniranja. Na početku poglavlja spomenuti su akronimi LMS i VLE. Ostali nazivi za slične koncepte su upravljano okruženje za učenje (engl. *Managed Learning Environment*, skraćeno MLE), sustav za podršku učenju (engl. *Learning Support System*, skraćeno LSS), umreženi centar za učenje (engl. *Online Learning Centre*, skraćeno OLC) i obrazovna platforma (engl. *Learning Platform*, skraćeno LP) [84]. Razlike u značenju i načinu funkcioniranja tih sustava sa stajališta s kojeg ih promatra ovaj rad su neznatne, tako da se u nastavku rada neće praviti stroge razlike među njima već će se prethodno navedenom definicijom VLE-a obuhvatiti svi slični sustavi.

2.3.2 Karakteristike virtualnog obrazovnog okruženja

VLE je dominantan oblik sustava za računalom podržano učenje u posljednjih petnaestak godina. Na njegov razvoj utjecala su tri važna faktora [88]. Spajanje i konsolidacija velikih proizvođača VLE-ova stvorilo je divove koji su postali dominantni na tržištu (npr. spajanje *WebCT* i *Blackboard* sustava provedeno 2005. godine). Standardizacija obrazovnih materijala (SCORM, LOM i IMS) omogućila je izradu i korištenje obrazovnih materijala unutar različitih okruženja. Izrađeni materijali više nisu zaključani za korištenje unutar određenog sustava već se mogu dijeliti i koristiti unutar različitih sustava. Treći faktor je znatno investiranje u razvoj otvorenih verzija VLE-ova od kojih su najpoznatiji *Moodle* [URI22] i *Sakai* [URI30]. Otvorena okruženja omogućuju veću prilagodbu zahtjevima korisnika jer institucija ima pristup njihovom izvornom kodu. Osim toga njihov razvoj je vođen od strane zajednice i svatko tko ima resurse i interes može se uključiti u njega. Takav pristup je utjecao na povećanje kvalitete tih okruženja tako da je danas ona usporediva s kvalitetom komercijalnih rješenja. Takva okruženja u većoj mjeri podržavaju konstruktivistički način učenja, a učenici dobivaju veću mogućnost samostalnog kreiranja obrazovnih sadržaja. Cjenovna komponenta nije zanemariva jer su komercijalna rješenja vrlo skupa, tako da sve više institucija za svoje potrebe koristi otvorene VLE-ove.

U istom razdoblju dogodio se rast popularnosti Web 2.0 koncepta i aplikacija koje omogućuju suradnju korisnika i mogu se koristiti u procesu učenja. Primjeri takvih aplikacija su blogovi,

wikiji i društvene mreže. Te aplikacije su brzo prihvaćene od strane mnogih korisnika weba i njihovi elementi se počinju uključivati unutar VLE-ova. U posljednjih nekoliko godina zapaža se pozitivan pomak na tom području njihovim uključivanjem u preporuke za korištenje obrazovnih materijala [104].

2.3.3 Opis virtualnog obrazovnog okruženja

Osnovna karakteristika VLE-ova je da su prilagođeni institucionalnom prijenosu znanja i u skladu s tim je kreirana i njihova struktura. VLE-ovi su modernizirali izvođenje nastave na tehnološkoj razini i učinili materijale i aktivnosti dostupnije učenicima, ali u svojoj osnovi oni i dalje podržavaju klasičan model učenja u razredu ili grupi gdje grupa studenata u isto vrijeme stječe znanja iz istog područja, po istom programu, učeći iz istih nastavnih materijala. Kao prije trideset godina u klasičnoj učionici, učenici se grupiraju, započinju učenje u isto vrijeme, periodički dobivaju materijale i očekuje se da svi završe učenje u isto vrijeme (u okviru semestra ili planiranog trajanja tečaja) [29]. Taj model VLE-a izabran je jer je bio najekonomičniji za korištenje unutar institucija. Korištenjem takvog modela nekolicina nastavnika ili instruktora može učinkovito podučavati veći broj učenika, zadavati im zadatke i provjeravati njihove rezultate [38]. VLE okruženje idealno je kako nadopuna klasičnoj nastavi u učionici jer omogućuje učenicima pristup nastavnim materijalima i sadržajima van termina nastave tako da posljednjih nekoliko godina sve više obrazovnih institucija uvodi VLE-ove kao nadopunu klasičnom obliku nastave u učionici.

2.3.4 Elementi virtualnog obrazovnog okruženja

VLE obuhvaća programsku podršku koja omogućuje potpuno administriranje procesa učenja i poučavanja u okruženju virtualnog razreda. U globalu se sve funkcije VLE-a mogu podijeliti u dvije skupine: administrativne i nastavne.

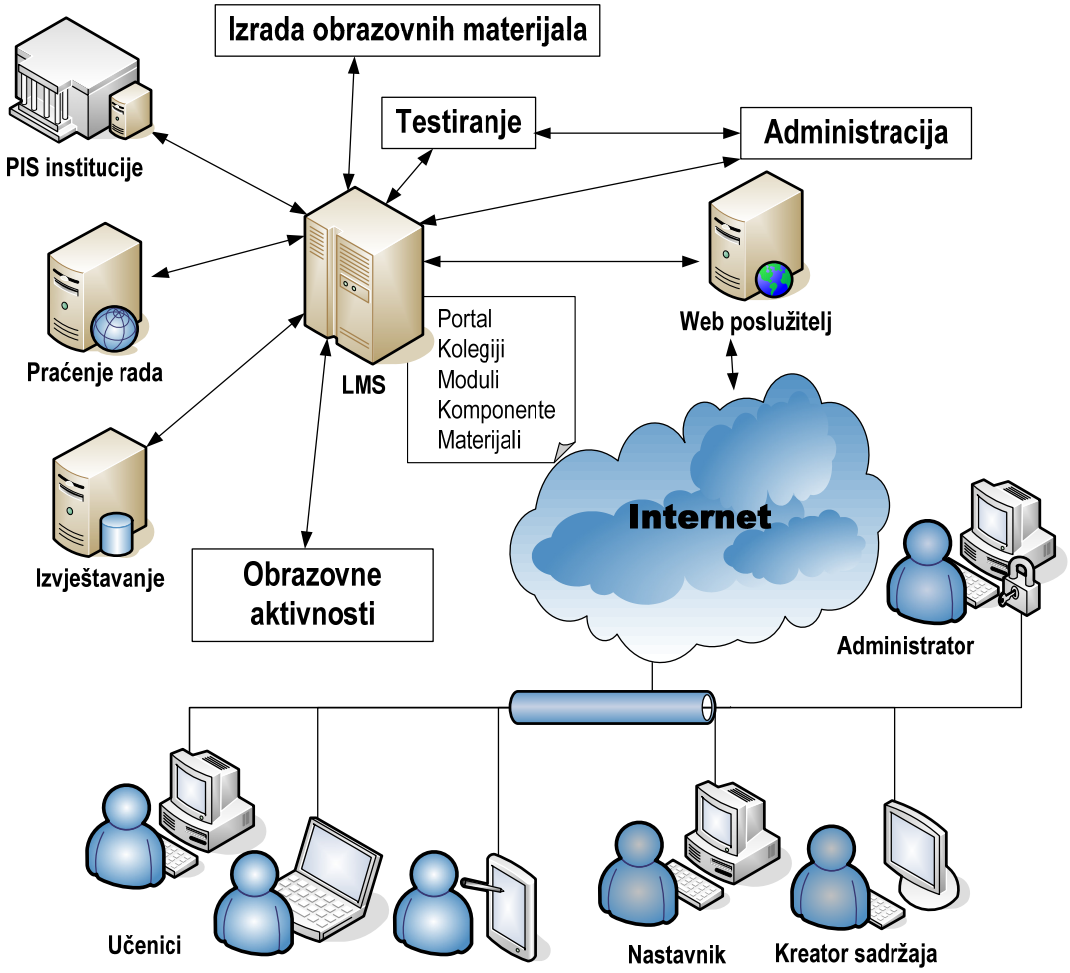
Administrativne funkcije obuhvaćaju sve evidencije podataka nužnih za funkcioniranje sustava, a koje nisu izravno vezane uz procese prijenosa znanja. Takve evidencije uključuju evidenciju korisnika sustava, učenika, kolegija, dozvola i korisničkih grupa, prijavu korisnika u kolegije, izvještavanje o pohađanju nastave, napretku, statusu i rezultatima učenja, podršku izradi nastavnih materijala i vezu s poslovnim informacijskim sustavom institucije.

Nastavne funkcije su izravno vezane uz prijenos znanja učenicima i obuhvaćaju pristup kolegiju i materijalima unutar njega, izvršavanje zadanih aktivnosti u sklopu kolegija, praćenje izvršenja aktivnosti i testiranje znanja. Sustav omogućuje izradu i postavljanje obrazovnih materijala koji su najčešće oblikovani u lekcije. Učenici prijelazom s jednog na drugi sadržaj postepeno stječu znanje. Slijedna struktura materijala nije obavezna ali se najčešće koristi. U adaptivnim okruženjima koja su u većoj mjeri orijentirana učenicima, oni mogu samostalno izabrati put kojim će prolaziti kroz sadržaje. Važno je jedino da pokažu stečeno znanje iz svih predstavljenih područja. Provjera znanja koja je implementirana unutar sustava, obično se provodi putem testova za samoprovjeru. Kod većine sustava obrazovni materijali su oblikovani kao web stranice unutar kojih se uključuju različiti multimedijalni sadržaji. Materijali se međusobno povezuju hipervezama čime se omogućuje jednostavno

kretanje po vezanim materijalima. Standardizacija obrazovnih materijala omogućila je korištenje obrazovnih materijala različitih proizvođača unutar VLE-a.

VLE-ovi sadrže brojne alate koji služe za obavljanje specifičnih funkcija planiranja, organiziranja i upravljanja procesom učenja. Najvažniji elementi VLE-a su ulazni portal, personalizirani učenički portal, katalog kolegija, baza podataka korisnika, sustav za praćenje i izvještavanje, sustav za isporuku nastavnih sadržaja i upravljanje nastavnim resursima te administrativne funkcije.

Na slici 2.2 prikazan je model VLE-a baziran na webu kao platformi kojem učenici pristupaju putem Interneta, pomoću web preglednika.



Slika 2.2 Struktura VLE sustava

Ulaz u VLE obično je u obliku portala na kojem su vidljive osnovne informacije i novosti o sustavu. Taj dio sustava najčešće je javno dostupan i za njega nije potrebna prijava, dok su drugi dijelovi sustava zatvoreni i dostupni isključivo registriranim korisnicima nakon prijave u sustav. Da bi mogli koristiti VLE, korisnici se trebaju autorizirati i prijaviti za rad sa sustavom. Svaka skupina korisnika po prijavi koristi personalizirano sučelje koje ovisi o ulozi korisnika unutar sustava. Za učenika, ulazni modul je obično u obliku učeničkog portala na kojem ima uvid u svoje osobne podatke, kolegije koje pohađa, informacije o novostima i

terminima različitih aktivnosti i obaveza, a omogućena mu je komunikacija s drugim korisnicima sustava.

Katalog kolegija je dio sustava s popisom svih kolegija koji se mogu upisati, a uz svaki kolegij vidljive su njegove osnovne informacije. U sustavima koji dozvoljavaju samoregistraciju, učenici mogu pregledavati kolegije i samostalno se prijaviti za pohađanje izabranih kolegija. U sustavima kod kojih nije dozvoljena samoregistracija, učenike u izabrani kolegij upisuje administrator ili voditelj kolegija. Uz katalog kolegija mogu se vezati financijski podaci i na takav način se VLE može izravno povezati s poslovnim informacijskim sustavom institucije.

Nakon prijave na određeni kolegij, učenik dobiva pravo pristupa virtualnoj učionici predstavljenoj web stranicom kolegija. Na toj stranici nalaze se obrazovni materijali i informacije o tijeku kolegija. Korištenjem alata na toj stranici učenik može komunicirati s nastavnikom i drugim učenicima koji pohađaju isti kolegij, provoditi aktivnosti za koje ima dozvolu te dobiti uvid u svoj uspjeh i rezultate napredovanja. Nastavne materijale kolegija uglavnom stvaraju instrukcijski dizajneri ili nastavnici. Kod sustava koji u većoj mjeri podržavaju konstruktivističke principe učenici imaju mogućnost samostalno stvarati nastavne materijale koji su dostupni i drugim učenicima istog kolegija. Najjednostavniji način za stvaranje takvih materijala je korištenjem suradničkih Web 2.0 aplikacija kao što su forum, wiki i blog. Ti alati su učenicima poznati iz web okruženja, tako da obično nemaju većih problema s korištenjem takvih aplikacija unutar VLE-a. Jedini nedostatak koji korisnicima može smetati je što verzije tih aplikacija koje su uključene unutar VLE-a imaju smanjenu funkcionalnost u odnosu na verzije dostupne na webu. To je svjesna odluka prilikom izrade sustava, a donesena je kako bi korisnici unutar VLE-a trošili što je moguće manje vremena na ovladavanje korištenjem alata.

Unutar VLE-a obično postoje alati koji omogućavaju izradu jednostavnijih sadržaja u obliku web stranica i ispita te uključivanje gotovih multimedijalnih sadržaja unutar stranice.

Informacije o učenicima, osim osnovnih korisničkih i kontakt podataka, sadrže podatke o kolegijima koje učenik pohađa, aktivnostima koje je obavljao i materijalima koje je stvorio tijekom rada. Važan dio VLE sustava je mogućnost izvještavanja o statusu i napredovanju učenika tijekom učenja. Podaci o rezultatima i postignutom uspjehu mogu pomoći nastavnicima da poboljšaju način provođenja nastave, a statistike o pristupu i načinu korištenja materijala mogu pomoći kreatorima sadržaja u ispravljanju potencijalnih problema i grešaka u materijalima, te u njihovom poboljšanju.

3 WEB 2.0

U ovom poglavlju opisat će se povijest nastanka weba, njegova transformacija iz klasičnog weba u novi Web 2.0 oblik, osnovne karakteristike i neke od najvažnijih skupina aplikacija koje karakteriziraju novi model.

3.1 Web kao platforma

3.1.1 Razvoj Web 2.0 modela

Broj korisnika weba danas se povećava velikom brzinom, najviše zbog povećanja broja korisnika društvenih Web 2.0 aplikacija. Procijenjeni broj korisnika kreće se oko 1,67 milijardi što obuhvaća oko 24,7% ukupnog svjetskog broja stanovnika. U razdoblju između 2000. i 2009. godine broj korisnika weba se povećao za preko četiri i po puta [106].

Web je izvorno zamišljen kao internetski servis na kojem će znanstvenici moći objavljivati svoje materijale i koji će biti dostupni za pretraživanje i pregledavanje svim zainteresiranim korisnicima. Nedugo zatim kompanije su počele prepoznavati komercijalne mogućnosti, a dominantni korisnici weba nisu bili znanstvenici već „obični“ ljudi. U to vrijeme struktura web stranica i način uređivanja sadržaja bili su preslikani iz klasičnih medija poput časopisa ili novina. Pisci su izrađivali materijale, a urednici su bili odgovorni za njihovo uobličavanje i objavljivanje. Bila je vidljiva razlika između stvaraoca i korisnika sadržaja. Informacije su se pohranjivale na web stranicama različitih organizacija, poduzeća i medijskih kuća koje su zarađivale prodajom reklamnog prostora. Privatne web stranice bile su vrlo rijetke jer nisu bile razvijene tehnologije i aplikacije za jednostavno objavljivanje sadržaja. Osnovna motivacija za stvaranje sadržaja na webu bila je navesti posjetitelje na kupnju reklamiranih proizvoda i usluga.

S vremenom je zbog povećanja broja kompanija koje su prodavale reklamni prostor došlo do zasićenja, a posljedica je bila da su takve kompanije sve teže ostvarivale dobit. Dio internetskih kompanija izabrao je drugačiji pristup, stvaranje sadržaja i aplikacija orijentiranih korisnicima bez suvišnih reklama. Izrađene aplikacije su omogućavale jednostavno korištenje i korisnicima davale mogućnost objave vlastitih materijala te interakciju s drugim korisnicima. Funkcionalnost se dobivala besplatno, a količina reklama bila je minimalna tako da su korisnici imali osjećaj da dobivaju pravu vrijednosti [75]. Pioniri takvog načina razmišljanja bile su kompanije *Flickr* [URI12] i *Delicious* [URI8] koje su svoj poslovni model bazirale isključivo na stvaranju informacija od strane korisnika [63]. Prijelomna godina u kojoj se dogodilo zasićenje i pad mnogih internetskih kompanija koje su poslovanje bazirale na klasičnom poslovnom modelu je 2001. U jesen te godine zapaženo je da kompanije koje su izabrale novi pristup izgradnje sadržaja orijentiranih korisnicima napreduju, a ostale stagniraju. Novi model poslovanja nazvan je Web 2.0 i on ne opisuje tehnološki napredak već promjene koje su nastale u načinu stvaranja i korištenja informacija.

3.1.2 Osnovne karakteristike Web 2.0 modela

Web 2.0 model je skup principa koji funkcioniraju u praksi i vidljivi su u svim uspješnim aplikacijama koje postoje na tržištu. Osnovne karakteristike tog modela su stvaranje sadržaja od strane velikog broja korisnika, njihovo povezivanje u interesne grupe te intenzivna komunikacija, suradnja i razmjena informacija unutar grupa [63]. Glavna platforma za koju se izrađuju aplikacije je web i za korištenje tih aplikacija dovoljno je imati web preglednik. Programska rješenja više nisu monolitna već se mogu putem programskog sučelja (engl. *Application Programming Interface*, skraćeno API) povezivati s drugim aplikacijama. Na takav način proširuju svoju funkcionalnost i povećavaju broj korisnika. Društvena komponenta postaje vrlo važna, a web okruženje postaje važan kanal za stvaranje i održavanje društvenih odnosa bez obzira na fizičku udaljenost. U takvom okruženju velik broj korisnika stvara i objavljuje sadržaje na način i u obliku koji im odgovara i nad njima imaju punu kontrolu. Kako bi se olakšalo korištenje funkcionalnosti aplikacija javlja se trend izrade web aplikacija s unaprijeđenim grafičkim sučeljima (engl. *Rich Internet Applications*, skraćeno RIA) koje web aplikacijama donose funkcionalnost vrlo sličnu stolnim aplikacijama.

3.1.3 Web servisi

Web se sastoji od mnoštva povezanih računala različitih strojnih platformi i operacijskih sustava na kojima rade programi pisani u različitim programskim jezicima. Izvorno zamišljen kao platforma koju će koristiti ljudi, s vremenom je prerastao u okruženje koje u jednakoj mjeri koriste i drugi računalni sustavi. Ako je poznato sučelje aplikacije, na korištenje njezine funkcionalnosti od strane drugih sustava ne bi smjela utjecati strojna platforma na kojoj se aplikacija fizički nalazi niti način konkretne implementacije. Aplikacije koje omogućuju takav način korištenja nazivaju se web servisi.

Web servisi su logički zaokružene programske cjeline s precizno definiranim sučeljem koje mogu korištenjem standardnih tehnologija komunicirati s drugim web aplikacijama [113]. Bazirani su na standardnim protokolima i normama, neovisni o računalnoj platformi i programskom jeziku u kojem su pisani, a pomoću njih je moguće povezivati različite sustave i aplikacije. Pristupa im se putem njihove adrese, pozivom javno dostupnih funkcionalnosti koje čine korisničko sučelje servisa. Sučelje opisuje metode koje se mogu pozivati, parametre koje metode primaju i povratnu vrijednost koju po završetku obrade trebaju vratiti. Po pozivu web servisa pozvana metoda započinje obradu koristeći primljene parametre, a po završetku obrade vraća rezultate.

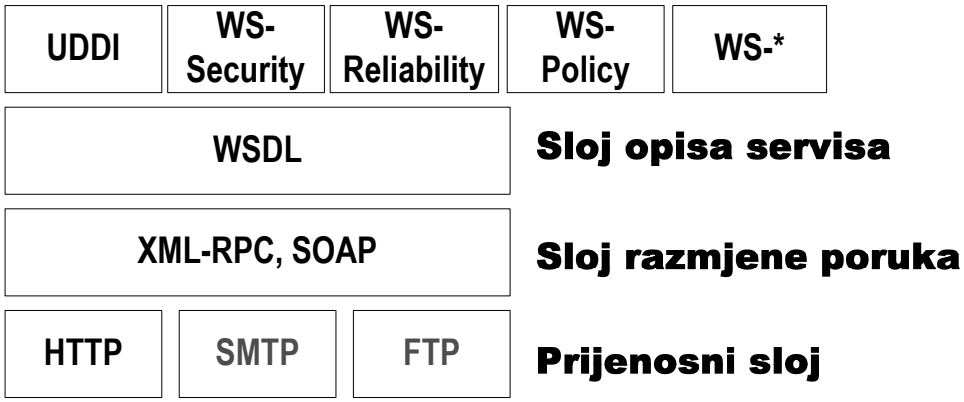
HTTP protokol koji se koristi na webu ima nekoliko metoda koje se mogu koristiti u komunikaciji. Najčešće korištene su GET i POST koje se svakodnevno koriste u okruženju web preglednika. Pomoću GET metode može se dohvatiti sadržaj bilo kojeg resursa smještenog na webu pomoću njegovog jedinstvenog identifikatora (engl. *Uniform Resource*

Identifier, skraćeno URI)[1], dok se POST metoda uglavnom koristi za prijenos podataka između klijenta i poslužiteljske aplikacije prilikom ažuriranja podataka. Ostale metode HTTP protokola su PUT i DELETE. Metoda PUT namijenjena je pohrani sadržaja na zadanu lokaciju. Ako sadržaj prethodno postoji on se treba ažurirati, a ako ne postoji potrebno je kreirati novi sadržaj. Metoda DELETE služi za brisanje sadržaja na zadanoj adresi [30][31].

Dva najčešće korištena arhitekturna stila pri korištenju servisa na webu su servisno orijentirana arhitektura (engl. *Service Oriented Architecture*, skraćeno SOA) i resursno orijentirana arhitektura (engl. *Resource-Oriented Architecture*, skraćeno ROA). SOA je danas dominantan način razvoja programa u distribuiranim poslovnim sustavima. Osnovni princip te arhitekture je dekompozicija programa u skup servisa koji se mogu koristiti višekratno i neovisno jedan od drugoga. Njihovim međusobnim povezivanjem mogu se stvarati nove aplikacije. ROA je izraz koji se koristi za opis razvoja programske podrške na webu izgrađene prema REST (*Representational State Transfer*) principima. REST opisuju način na koji se pristupa mrežnim resursima putem njihovog URI-ja, korištenjem standardnih metoda HTTP protokola [30]. Povezivanjem funkcionalnosti resursa izloženih na takav način moguće je izraditi složenu programsku podršku.

3.1.4 Servisno orijentirana arhitektura

Na slici 3.1 prikazana su četiri sloja protokola koji se mogu prepoznati kod servisno orijentirane arhitekture.



Slika 3.1 Slojevi protokola SOA web servisa

Za prijenos podataka koriste se protokoli aplikacijskog sloja na Internetu. Najčešće se koristi HTTP protokol, mada specifikacija dozvoljava i korištenje drugih protokola. Sloj razmjene poruka predstavljen je protokolima koji služe za oblikovanje omotnice pri prijenosu poruka. Ti protokoli su bazirani su na XML-u i obuhvaćaju jednostavniji XML-RPC i složeniji SOAP protokol. Sloj opisa servisa definira sučelje za pristup funkcionalnostima web servisa koje se opisuje pomoću WSDL (*Web Service Definition Language*) protokola. Na višim slojevima nalazi se protokol koji se koristi pri objavljivanju i pronalaženju web servisa (engl. *Universal*

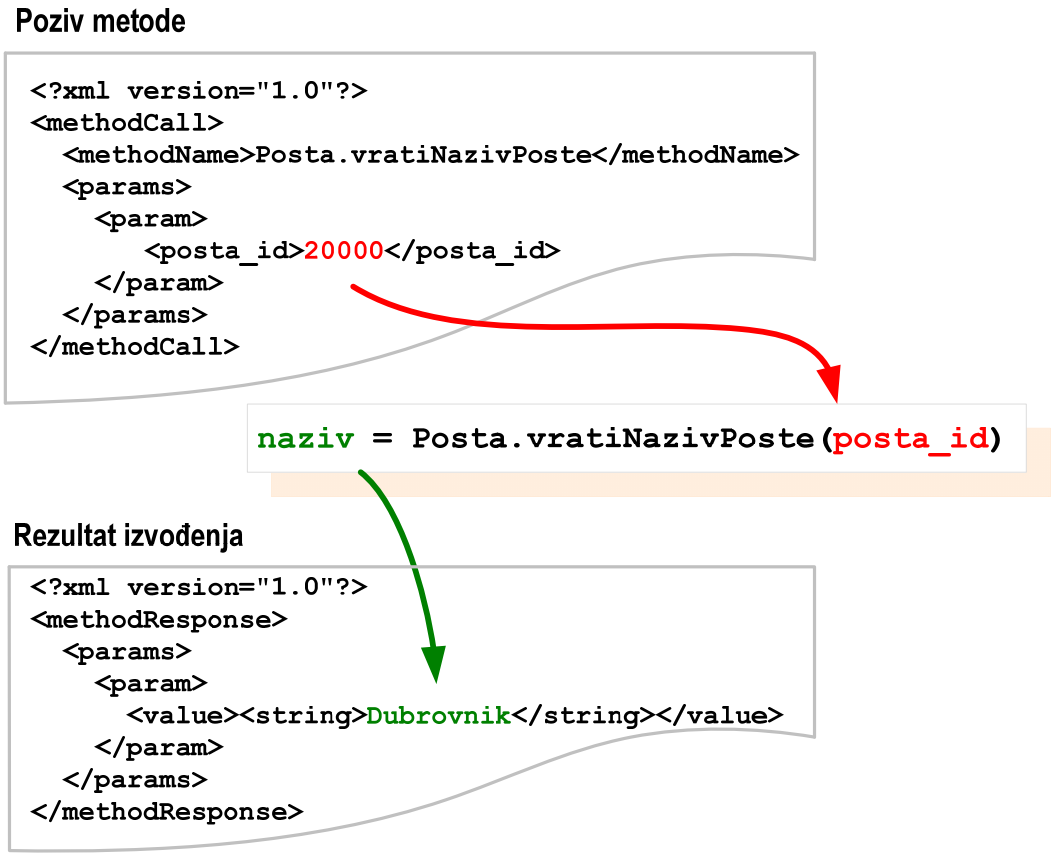
[1] Na webu se često kao sinonimi koriste URI i URL (engl. *Uniform Resource Locator*) premda među njima postoji razlika. URL pokazuje na konkretnu lokaciju na kojoj se sadržaj nalazi, dok URI jednoznačno imenuje sadržaj. Svaki URL je istovremeno URI, ali svaki URI ne pokazuje lokaciju resursa.

Description Discovery and Integration, skraćeno UDDI). Protokoli sloja razmjene poruka unutar sebe nemaju implementirane sigurnosne mehanizme pa je za rješenje tog problema razvijen niz protokola koji se često zajedničkim imenom nazivaju WS-* standardi.

3.1.4.1 XML-RPC

Ideja udaljenog pozivanja procedura (engl. *Remote Procedure Call*, skraćeno RPC) bazira se na korištenju funkcionalnosti računalnih programa smještenih na udaljenoj mrežnoj lokaciji kao da se radi o procedurama unutar lokalnog programa. Korištenjem RPC-a klijent komunicira s poslužiteljem slanjem podataka u zatvorenoj omotnici adresiranoj na URI servisa. Omotnica sadrži naziv procedure koja se poziva i potrebne vrijednosti parametara. Na osnovu primljenog zahtjeva poslužitelj vrši obradu i vraća rezultate zapakirane u sličnoj omotnici. XML-RPC je protokol za udaljeno pozivanje procedura na webu razmjenom poruka u tekstualnom XML formatu koji je prikladan za korištenje pomoću HTTP protokola. Sve metode izabranog web servisa imaju isti URI jer se nazivi metoda navode unutar poruke. XML-RPC protokol koristi HTTP POST metodu [89].

Na slici 3.2 prikazan je primjer udaljenog pozivanja metode korištenjem XML-RPC protokola.



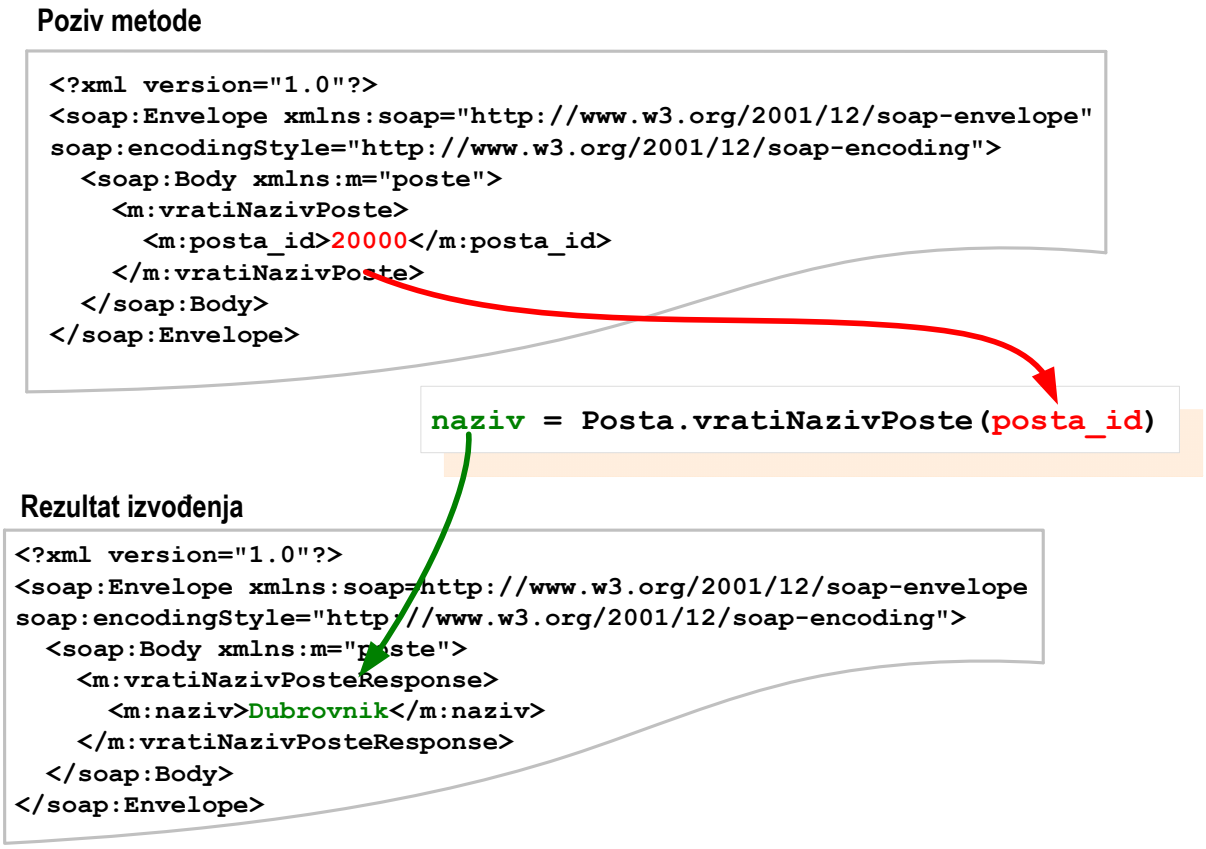
Slika 3.2 Primjer korištenja XML-RPC protokola

3.1.4.2 SOAP

SOAP protokol nastao je razvojem i unaprjeđenjem XML-RPC protokola za primjene u okruženju poslovnih sustava. Osim funkcionalnosti pozivanja udaljenih procedura, SOAP protokol nudi i funkcionalnost prijenosa objekata koji je posebno prikladan za korištenje u poslovnim sustavima jer podržava objektni način razmišljanja o sustavu. Kod takvog načina prijenosa, poslovni objekt se zapakiran unutar SOAP omotnice prosljeđuje drugom sustavu.

Davatelj servisa korištenjem WSDL protokola opisuje sučelje (dostupne metode, parametre, tipove podataka i povratne vrijednosti) koje se mogu koristiti. Nakon što korisnik dohvati WSDL opis i iz njega sazna informacije o sučelju, nastavak procesa korištenja identičan je opisanom kod XML-RPC protokola. Jedina razlika je u formatu poruke koja koristi oblik definiran SOAP protokolom.

Na slici 3.3 prikazano je udaljeno pozivanje procedure korištenjem SOAP protokola.



Slika 3.3 Primjer razmjene poruka SOAP protokolom

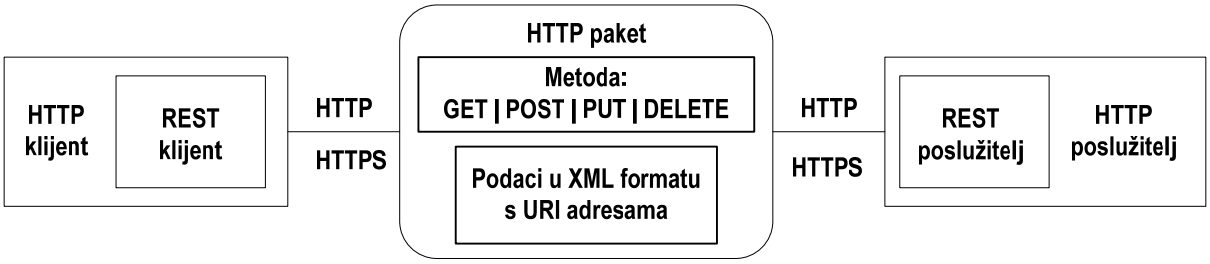
3.1.5 Resursno orijentirana arhitektura

3.1.5.1 REST

REST nije protokol već stil koji se koristi pri razvoju aplikacija u distribuiranim sustavima. U mrežnom okruženju postoji mnoštvo resursa u digitalnom obliku (npr. web stranica, slika, pdf dokument ili početna stranica nekog web portala) koji se mogu jednoznačno imenovati

pomoću URI-ja. Sadržaj resursa s vremenom se može promijeniti, ali njegovo značenje i adresa ostaju isti [30]. REST nije vezan niti uz jedan konkretan protokol, ali sve praktične implementacije vezane su uz HTTP protokol [66].

Slika 3.4 prikazuje osnovnu arhitekturu REST sustava u kojim klijenti korištenjem HTTP protokola pristupaju resursima smještenima na web poslužiteljima.

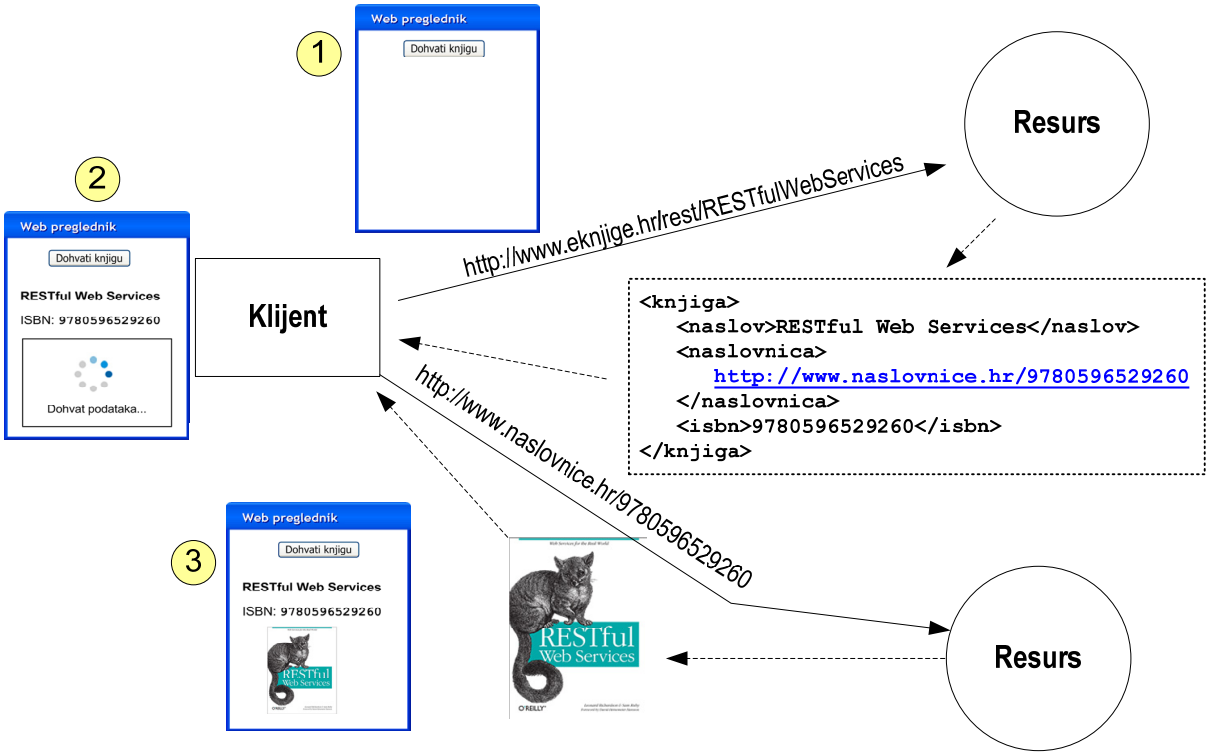


Slika 3.4 REST arhitektura

Web se može promatrati kao okruženje sastavljeno od samostalnih (engl. *stateless*) klijenata i poslužitelja. Resursi su smješteni na poslužiteljima i izloženi dohvatu od strane klijenata putem svog URI-ja, korištenjem HTTP protokola. Za dohvat i obrade nad resursima, klijenti koriste sve dostupne HTTP metode: GET, POST, PUT i DELETE. Kada klijent želi dohvatiti reprezentaciju određenog resursa, to će učiniti pomoću GET metode, a želi li stvoriti novi resurs, koristiti će POST metodu. Izmjena postojećeg resursa moguća je pomoću PUT metode, a za brisanje nepotrebnog resursa koristi se DELETE metoda.

Da bi se saznalo koju akciju klijent želi izvršiti, dovoljno je pročitati prvu liniju HTTP zahtjeva tako da je moguće pozivati web servise korištenjem samo jedne linije protokola [66]. Npr. linija `GET /izvjesca/prijavljene-greske HTTP/1.1` znači da se žele dohvatiti trenutno izvješće o prijavljenim pogreškama u radu sustava koje je smješteno na pozvanom poslužitelju u direktoriju `izvjesca`. Prilikom slanja odgovora, REST servis vraća standardne HTTP kodove, pa će npr. ako je akcija čitanja bila uspješna vratiti kod 200, a ako se pri dohvatu podataka dogodila neka pogreška vratiti će grešku iz raspona 300 do 599 [31].

Na slici 3.5 prikazan je primjer komunikacije korištenjem REST pristupa.



Slika 3.5 Primjer REST komunikacije

Osoba želi saznati informacije o knjizi *RESTful web services* koje su objavljene na webu. Reprezentacija resursa je u ovom slučaju datoteka s podacima o knjizi u XML formatu smještena na web stranici. Putem aplikacije u web pregledniku osoba pristupa ciljnom URI-ju. Web poslužitelj obrađuje upit i vraća reprezentaciju traženog resursa. Po prijemu reprezentacije resursa, klijent prelazi u odgovarajuće stanje koje u ovom slučaju znači da je primio osnovne podatke o knjizi, te da je potrebno dohvatiti sliku naslovne stranice knjige. URI tog resursa se nalazi među primljenim podacima. Klijentska aplikacija pristupa novom resursu koristeći taj URI. Web poslužitelj vraća reprezentaciju tražene slike u grafičkom obliku. Prijem slike klijenta prebacuje u novo stanje, a na zaslonu web preglednika se prikazuje primljena slika. Ako dohvat slike ne bi bio uspješan, klijent bi prešao u novo stanje, a unutar aplikacije bi se korisniku prikazala informacija da traženi resurs nije dostupan. Općenito klijent mijenja stanje s prijemom svake nove reprezentacije resursa.

Izraz REST danas je vrlo popularan i često se koristi za opis bilo kojeg jednostavnog sučelja za prijenos podataka putem HTTP protokola koje ne koristi XML-RPC ili SOAP protokol. Mnoge praktične implementacije su hibridi između REST i XML-RPC modela koji se ponašaju u skladu s REST definicijom prilikom dohvata podataka, a podatke stvaraju, ažuriraju i brišu korištenjem XML-RPC modela. To dovodi do nesporazuma u komunikaciji o tehnologiji koju web servisi koriste, pa je zbog toga uobičajeno sustave koji su rađeni u skladu sa striktnim REST principima nazivati RESTful sustavi. U takvim sustavima svaki resurs dostupan je putem jedinstvenog URI-ja, prijenos podataka obavlja se razmjenom

tekstualnih poruka korištenjem XML ili JSON protokola, a za operacije se koriste sve dostupne HTTP metode [66].

3.1.6 Hibridne web aplikacije

Web servisi kao tehnologija predstavljaju korak prema labavo povezanoj strukturi weba (engl. *loosely connected*). Kako bi podržali takav razvoj sve više web aplikacija omogućuje korištenje dijela svoje funkcionalnosti drugim sustavima pretvarajući se u web servise. Dostupnu funkcionalnost opisuju API-jem i time omogućavaju stvaranje hibridnih web aplikacija. Osnovna osobina takvih aplikacija je omogućavanje pristupa informacijama iz različitih izvora bez potreba za odlaskom na sve web stranice na kojima se te informacije izvorno nalaze te njihovo predstavljanje u novom obliku [28][91]. Njihovim korištenjem podaci se mogu predstaviti na novi način ili se mogu stvoriti potpuno nove informacije.

Po funkcionalnosti hibridne web aplikacije se mogu podijeliti u dvije skupine. Prvoj pripadaju one koje unutar zajedničkog sučelja prikupljaju podatke iz više različitih izvora pomoću RSS (*Really Simple Syndication*) *feedova* ili veza na web sadržaje putem URI-ja. Primjer takvih aplikacija su osobni web portali bazirani na *Ajax* (*Asynchronous JavaScript and* XML) tehnologiji [URI18][URI26][URI27] koji koriste funkcionalnosti gotovih *widgeta*. Većina hibridnih aplikacija koja se danas koriste izrađena je na ovakav način jer je takav način stvaranja sadržaja prikladan za krajnje korisnike. Drugoj skupini pripadaju složenije aplikacije koje kombiniraju podatke iz različitih izvora stvarajući nove informacije. Takva vrsta aplikacija zahtjeva programiranje korištenjem API-ja, za njihovo stvaranje potrebna su specifična programerska znanja i zbog toga nisu pogodne za krajnje korisnike [15]. Da bi se pojednostavnilo korištenje web servisa u hibridnim aplikacijama krajnjim korisnicima, predlaže se opis njihove semantika [71] na principu semantičkog weba [7].

3.1.7 *Widgeti*

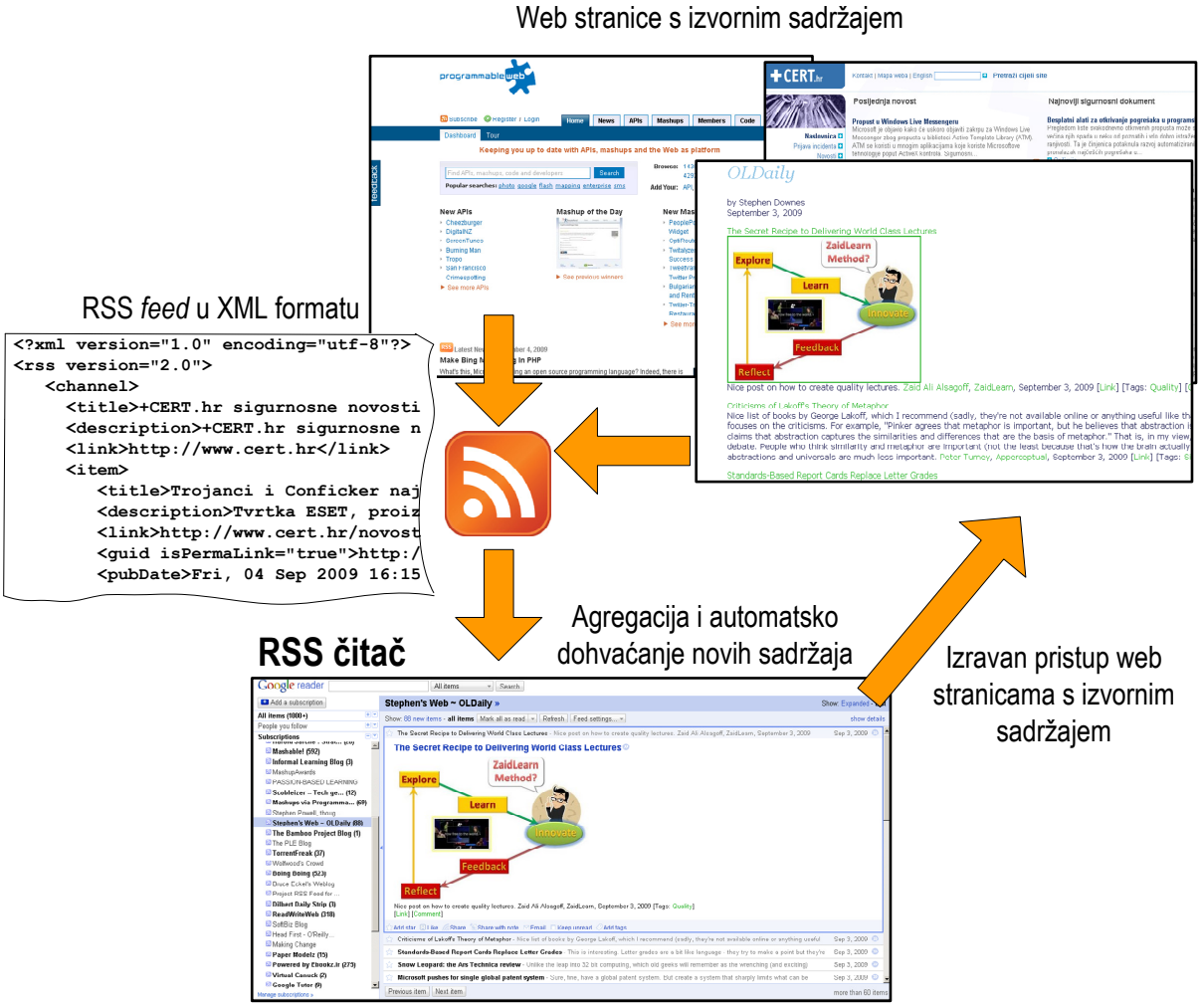
Hibridne web aplikacije se često izrađuju povezivanjem samostalnih mini aplikacija zaokruženih funkcionalnosti (engl. *widget*). *Widgeti* se koriste za razne namjene počevši od jednostavnog prikazivanja sadržaja druge web stranice, RSS sadržaja, slika, videa, pa sve do aplikacija sa specifičnom funkcionalnošću poput programa za pretvaranje valuta. Obično se umeću u web stranicu navođenjem HTML ili *JavaScript* (skraćeno JS) programskog koda. Termin *widget* se ne odnosi na funkcionalnost aplikacije već na način na koji se ona čini dostupnom, a to je u obliku male forme zaokružene funkcionalnosti. Drugi često korišteni nazivi za isti koncept su gadgeti (engl. *gadget*) ili programčići (engl. *snippet*) [91]. *Widgeti* se mogu koristiti na webu i u sklopu stolnih aplikacija, a u nastavku ovoga rada, pod pojmom *widgeti* podrazumijevati će se *widgeti* koji se koriste na webu.

3.1.8 RSS

Karakteristika današnjeg weba je trend neprekidnog rasta količine informacija koje se objavljuju i postaju dostupne putem mreže. Tome u velikoj mjeri pridonose informacije koje se objavljuju na blogovima i wiki sustavima. RSS je obitelj web formata koje koriste web

stranice koje često mijenjaju sadržaj, kako bi novi sadržaj odmah po objavi učinile dostupnima drugim korisnicima. Korist od RSS dokumenata imaju izdavač i korisnici. Izdavači objavom sadržaja sa svog weba osiguravaju ažurno obavještavanje zainteresiranih korisnika o novostima, a korisnici se putem RSS čitača pretplaćuju na izabrane RSS sadržaje i na takav način osiguravaju ažuran dotok informacija.

Na slici 3.6 prikazan je način na koji funkcionira korištenje RSS sadržaja.



Slika 3.6 Opis funkcioniranja RSS-a

Kada pronađu sadržaj koji ih zanima, iz okruženja RSS čitača korisnici mogu pristupiti izvornom sadržaju koji se nalazi na web stranici izdavača. RSS čitač po zahtjevu korisnika ili samostalno u odgovarajućim vremenskim intervalima provjerava da li se dogodila promjena sadržaja na izdavačevoj web stranici, i ako se dogodila preuzima nove materijale. RSS dokument (engl. RSS *feed*) obično sadrži naslov objavljenog članka, njegov sažetak i vezu na izvorni dokument.

RSS je samo jedan od nekoliko sličnih protokola koji se koriste za istu namjenu. Svi oni dijele isto polazište, a osim RSS-a popularni su još i *Atom* i *RDF Site Summary* protokol. U svima njima sadržaji se objavljuju u XML formatu.

3.2 Društvene mreže

Prema postavkama teorije društvenih mreža koja promatra društvene odnose među ljudima, veze između osoba mogu se prikazati u obliku grafa sastavljenog od međusobno povezanih čvorova. Čvorovi predstavljaju pojedince unutar mreže, a veze mogu biti različitih vrsti počevši od obiteljskih, prijateljskih i emocionalnih preko interesnih do profesionalnih.

3.2.1 Teorija društvenih mreža

Teoriju je prvi formulirao sociolog J. Barnes 1954. godine kako bi opisao ulogu pojedinca u društvu [41], iako je sam koncept društvenih mreža poznat odavno. Jedan od očitijih primjera mreža koje su odavno dobro funkcionirale u praksi je rad znanstvenika na nekom području. Znanstvenici proučavaju radove drugih znanstvenika koji rade na istom području, povezuju se s njima, referenciraju na njihove radove i proširuju ukupna dostignuća.

Teorija je svoju praktičnu potvrdu našla u okviru Web 2.0 koncepta. U stvarnom okruženju teško je biti član mnogih društvenih mreža jer osobni kontakti zahtijevaju velik utrošak vremena. Osim toga osobe su ograničene na društvene mreže u bližem fizičkom okruženju. Na webu udaljenost više nije problem i osoba može sudjelovati u društvenim mrežama neovisno o mjestu boravka i vremenskoj zoni. Prilikom traženja i upoznavanje sličnih osoba većinu posla će obaviti pretraživač na osnovu osobnih podataka koje su članovi mreže zapisali o sebi. Danas gotovo svaka Web 2.0 aplikacija omogućuje stvaranje društvenih mreža orijentiranih na odgovarajuće područje.

Na popularizaciju društvenih mreža na webu dosta je utjecala teorija o šest stupnjeva odvojenosti koja tvrdi da svaka osoba može doći u kontakt s bilo kojom drugom živućom osobom preko maksimalno 6 drugih osoba. Teoriju je šezdesetih godina 20. stoljeća znanstveno osmislio psiholog Stanley Milgram provevši eksperiment doslovce slijedeći tu priču [5]. Pokušao je stupiti u kontakt s dva nasumce izabrana čovjeka u SAD-u na način da je poslao pisma zamolbe nasumce odabranim osobama. Pravilo kojeg su se svi sudionici u lancu morali pridržavati je da slijedeću osobu u lancu poznaju osobno. Eksperiment je pokazao da je prosječna udaljenost između dvije osobe u SAD-u u to vrijeme bila oko 5,5. Ispravnost provedenog eksperimenta je u više navrata dovedena u pitanje, no sama teorija je zainteresirala mnoge. Slično istraživanje koje je 2007. godine provela tvrtka *Microsoft* na 240 milijuna korisnika programa *Microsoft Messenger* potvrdilo je teoriju došavši do stupnja odvojenosti 6,6 [48]. Bez obzira na konkretne brojke, broj stupnjeva odvojenosti između dvije osobe je iznenađujuće mali s obzirom na broj stanovnika na planetu i ta spoznaja utječe na korisnike društvenih mreža na webu.

3.2.2 Društvene mreže u Web 2.0 aplikacijama

Web aplikacije koje sadrže koncept društvenih mreža mogu se podijeliti u dvije skupine. Prvoj pripadaju aplikacije s osnovnom funkcijom povezivanja. Dvije najveće mreže takve vrste su *Facebook* [URI11] i *MySpace* [URI25], koje su namijenjene okupljanju opće kategorije korisnika. *LinkedIn* [URI20] mreža pripada istoj vrsti aplikacija, a orijentirana je na specifičnu skupinu korisnika zainteresiranih za povezivanje po profesionalnoj osnovi.

Aplikacije toga tipa unutar sebe sadrže alate koji omogućavaju komunikaciju korisnika, zabavu i suradnički rad.

Drugoj skupini pripada većina ostalih Web 2.0 aplikacija čija osnovna namjena nije stvaranje društvenih mreža, ali funkcioniraju na tom principu. One okupljaju korisnike koje povezuju zajednički interesi. Tako se npr. oko *Flickr* aplikacije okupljaju korisnici zainteresirani za pohranu, razmjenu i pregledavanje slika, a oko *YouTube* [URI43] korisnici koji su zainteresirani za izradu, objavljivanje i pregledavanje video materijala.

Prema istraživanju navika korisnika Interneta provedenog 2008. godine, 58% aktivnih korisnika Interneta je član barem jedne društvene mreže. Što se tiče njihovih navika i aktivnosti, 74% ih koristi kao komunikacijski kanal s drugim osobama, 55% za objavljivanje i razmjenu fotografija, 23% za objavljivanje i razmjenu video sadržaja, dok 21% korisnika koristi različite aplikacije u sklopu društvenih mreža. [105].

3.2.3 Folksonomija

U većini web aplikacija autori imaju mogućnost na web poslužitelj postaviti svoje materijale i opisati ih tekstom i korisničkim oznakama (engl. *tags*). Posjetitelji servisa pregledavaju sadržaje prema njihovom opisu i pridruženim korisničkim oznakama. Korisničke oznake su nestrukturirani meta podaci kojima autori označavaju materijale u okruženju društvenih mreža kako bi ih stavili u odgovarajući kontekst i omogućili im lakši pristup. Takav način označavanja sadržaja brzo je postao popularan u okruženju društvenih mreža jer korisnicima daje potpunu slobodu pri označavanju. Nastala mreža web sadržaja označenih neformalnim oznakama koje korisnici sami slobodno kreiraju, obično se naziva folksonomija (engl. *folksonomy*) [83]. Folksonomija podatke ne organizira hijerarhijski već linearno u jednoj razini. Uvidom u frekvenciju pojavljivanja pojmova korisnici imaju okvirnu informaciju o njihovoj popularnosti. Čest je grafički prikaz folksonomije u kojem se popularniji pojmovi prikazuju većim slovima, a manje popularni manjim. Takav način prikazivanja se naziva oblak korisničkih oznaka (engl. *tag cloud*).

Osnovni problem folksonomije odnosi se na preciznost označavanja, a posljedica je subjektivnosti osobe koja označava. Osobe imaju različit pogled na svijet i dolaze iz različitih sredina (npr. neki su više tehnički, a drugi više društveno orijentirani). Ono što je neki korisnik označio oznakom *usluga*, ne mora nužno svakom korisniku značiti isto. Računarac će npr. tom oznakom označiti web stranice koje opisuju protokole web servisa dok će službenik uz taj pojam vezati kvalitetu usluge koju pruža stranci. S druge strane velika prednost folksonomije je ta što je kreirana od strane korisnika sustava, a ne vanjskih autoriteta i ona se u praksi koristi. Koristeći folksonomiju korisnik će ponekad dobiti informaciju koju nije tražio, no u većini slučajeva će lakše i preciznije pronaći traženu informaciju na takav način, nego što u slučaju korištenja klasičnih pretraživača.

3.3 Web aplikacije s unaprijeđenim korisničkim sučeljem

Kako bi se korisnicima olakšalo korištenje web aplikacija, teži se tome da korisničko sučelje bude što sličnije sučelju stolnih aplikacija. Takve aplikacije se nazivaju RIA. Cilj njihovog

razvoja je funkcionalnošću se približiti stolnim aplikacijama, a pri tome sačuvati dobre osobine web okruženja. Zbog intenzivnog razvoja na tom polju danas su stolne i RIA aplikacije sve sličnije po svom izgledu i funkcionalnostima.

3.3.1 Osobine RIA

3.3.1.1 Unaprijeđeno korisničko sučelje

Primjena RIA koncepta teži poboljšanju ergonomskih osobina web aplikacija i olakšavanju rada korisnicima. Istovremeno dizajnerima daje veće mogućnosti za izradu dopadljivijih sučelja koja će privući dodatne korisnike. Za razliku od klasičnih web aplikacija kod kojih je za prikaz specifičnih multimedijalnih podataka (npr. videa) potrebno unutar web preglednika koristiti specifične vanjske dodatke (engl. *plugin*), dio RIA tehnologija te podatke može prikazivati izravno. Po svojoj funkcionalnosti RIA-e su prvenstveno web aplikacije, i kao posljedica toga u njih se ugrađuju ograničenja koja ne dozvoljavaju pristup lokalnim resursima na računalu klijenta.

3.3.1.2 Održavanje stanja

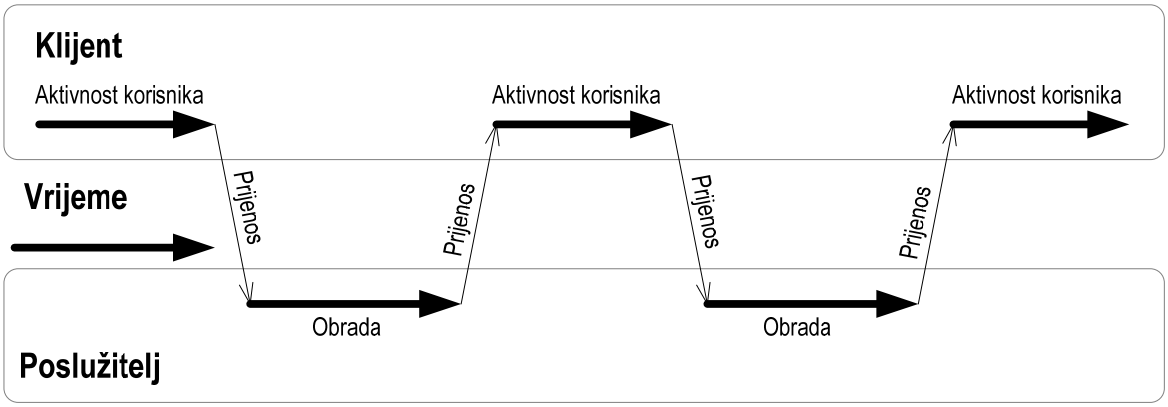
Većina suvremenih stolnih aplikacija pamti informacije o stanju prilikom prethodnog pokretanja pa aplikacija može povezati korisnika s njegovim prethodnim aktivnostima. Npr. nakon prijave korisnika, aplikacija može automatski učitati dokument na kojem je prošli put radio i pozicionirati ga na posljednji korišteni odlomak. Takav način pojednostavljuje rad pa su slične funkcionalnosti poželjne i u web aplikacijama.

Web je baziran na HTTP protokolu koji ima osobinu da su klijenti koji se vežu na poslužitelj samostalni. Da bi poslužitelj mogao povezati višestruke upite istog klijenta, programeri moraju u razvoju aplikacija koristiti različite trikove poput pamćenja sesije ili korištenja web kolačića (engl. *cookie*). RIA tehnologija unutar sebe rješava taj problem pa programer prilikom razvoja ne mora voditi računa o održavanju veza između klijenta i poslužitelja već se o tome brine sama platforma.

3.3.1.3 Asinkrona komunikacija

Korisnik putem korisničkog sučelja na klijentu unosi podatke i šalje ih poslužitelju. Nakon što poslužitelj primi zahtjev započinje proces obrade, a po završetku rezultate vraća klijentu. Za čitavo vrijeme dok traju prijenos i obrada na poslužitelju, korisnik čeka rezultate i ne može ništa drugo raditi.

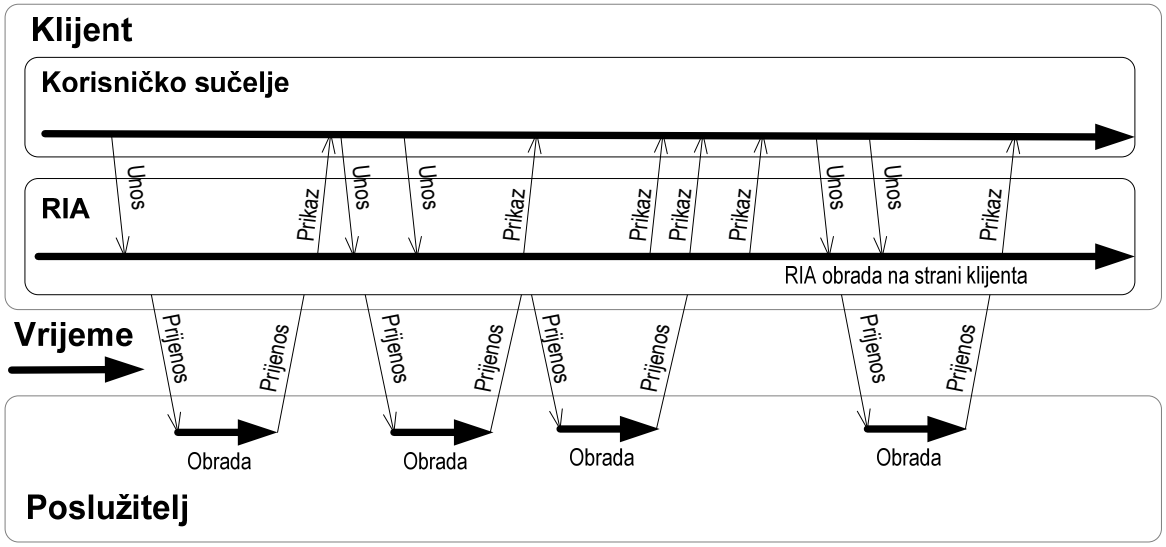
Klasičan sinkroni model funkcioniranja web aplikacije prikazan je na slici 3.7.



Slika 3.7 Sinkroni model funkcioniranja web aplikacije

U web okruženju gdje se mogu dogoditi zagušenja mreže i zastoji u komunikaciji, takav način rada nije optimalan jer se događa mnogo praznoga hoda.

U takvim okruženjima korisniji je asinkroni pristup komunikaciji, prikazan na slici 3.8.



Slika 3.8 Asinkroni model funkcioniranja RIA

Pretpostavka za asinkronu komunikaciju je da poslužitelj zna prepoznati svakog klijenta s kojim je povezan. Klijent može u bilo kojem trenutku zatražiti uslugu od web poslužitelja. Nakon što je korisnik putem sučelja unio podatke i poslao zahtjev za obradom, može nastaviti s drugim aktivnostima. Podatke preuzima međusloj klijentske RIA zadužen za koordinaciju asinkrone komunikacije. RIA međusloj prosljeđuje zahtjev za obradom web poslužitelju. Web poslužitelj zaprima zahtjev, a obraditi će ga i isporučiti rezultate onda kada to njegovi resursi dozvole. Po završetku obrade poslužitelj vraća rezultate klijentu. Podatke preuzima RIA međusloj koji koordinira njihov primitak i prosljeđuje ih za prikaz na korisničkom sučelju.

Takav način rada poboljšava tijek komunikacije između klijenta i poslužitelja i ubrzava rad jer se ne javljaju prazni hodovi uzrokovani čekanjem na prijenos podataka i završetak obrade na

poslužitelju. Osim toga prilikom promjene nekog od podataka na web stranici, više nije potrebno osvježavati čitavu stranicu već je dovoljno osvježiti samo onaj dio na kojem se nalaze promijenjeni podaci. Npr. ako su se promijenili samo podaci u tablici dovoljno je osvježiti tablicu, a svi ostali dijelovi web stranice ostaju nepromijenjeni. Statični podaci koji se ne mijenjaju prenijeti će se samo jednom, a više puta će se prenositi samo podaci koji su se tijekom obrade promijenili. Time se smanjuje količina podataka u prijenosu između klijenta i poslužitelja kao i broj konekcija, a korisnici imaju osjećaj da aplikacija brže radi.

3.3.1.4 Izvođenje na raznim platformama

Važna osobina koju svaka RIA treba zadovoljiti je izvođenje na različitim strojnim platformama i unutar različitih operacijskih sustava. Prema okruženju u kojem se izvode, RIA aplikacije se mogu podijeliti u dvije skupine. Prvoj pripadaju one koje se izvršavaju u okruženju web preglednika bez korištenja vanjskih dodataka. Takva tehnologija poznata je pod imenom *Ajax* i u najvećoj mjeri je utjecala na popularnost korištenja RIA koncepta.

Drugoj skupini pripadaju rješenja koja se izvode u okruženju virtualnih strojeva odnosno vanjskih dodataka unutar web preglednika. Ta rješenja su znatno popularnija pri izradi dijelova web aplikacija, ali ne i kompletnih aplikacija. Osnovni razlog je što su podaci unutar takvih aplikacija teško dostupni za indeksiranje web pretraživačima [57], iako su u posljednjih nekoliko godina napravljeni znatni pomaci u kvaliteti indeksiranja takvih sadržaja [1].

3.3.2 RIA tehnologije

3.3.2.1 Samostalne RIA aplikacije

Vanjski dodaci koji se koriste u okruženju web preglednika za izvođenje RIA veličinom trebaju biti mali. Više funkcionalnosti ukomponiranih u taj vanjski dodatak povećati će njegovu veličinu. Pored toga aplikacije koje se koriste u okruženju web preglednika imaju uključene posebne sigurnosne mehanizme (npr. ne mogu pristupati lokalnim resursima računala). Zbog tih problema, u posljednje vrijeme se počela razvijati tehnologija RIA koje se ne izvode unutar web preglednika, već su to samostalne aplikacije unutar operacijskog sustava. Primjer takve tehnologije je AIR (*Adobe Integrated Runtime*). AIR platforma omogućuje izradu internetskih aplikacija za različite operacijske sustave korištenjem istih tehnologija kao kod klasičnih *Flash* RIA koje se izvode u okruženju *Flash playera*. Osnovna razlika je što je AIR aplikacija samostalna aplikacija koje se izvodi na operacijskom sustavu. Nedostatak takvog pristupa je što je aplikaciju potrebno instalirati na računalo. S druge strane pošto se radi o samostalnoj aplikaciji korisnici imaju veće mogućnosti i nisu ograničeni veličinom virtualnog stroja. Korištenjem takvih aplikacija omogućen im je pristup resursima lokalnog računala na koje je aplikacija instalirana, postavke se više ne moraju spremati isključivo na web poslužitelj, a tijekom rada podaci se mogu spremati lokalno na računalo. Samim time omogućen je rad s aplikacijom i kada računalo nije vezano na Internet [37]. Korisnik se može spojiti na Internet, preuzeti podatke s poslužitelja i prekinuti vezu. Nakon toga u lokalnom okruženju AIR aplikacije može obrađivati preuzete podatke. Po završetku obrade može se ponovno spojiti na Internet i prenijeti poslužitelju rezultate obrade. Veza na

Internet nije potrebna osim za prijenos podataka između klijenta i poslužitelja, a sva obrada obavljena je *offline*. Takav način rada prikladan je u okruženjima u kojima se korisnici na Internet spajaju putem tehnologija baziranih na mobilnom internetu (GPRS, UMTS i sl.) jer je cijena prijenosa podataka kod njih još uvijek relativno veća u odnosu na klasično spajanje.

Po načinu funkcioniranja nema velike razlike između takvih aplikacija i klasičnih Java ili .NET aplikacija, tako da se prednost korištenja te tehnologije može očitovati kod razvojnih timova koji imaju iskustvo u razvoju klasičnih RIA pa ne moraju učiti nove tehnologije.

3.3.2.2 RIA tehnologije bazirane na web pregledniku

Ajax je skup tehnologija (HTML, CSS, JS, XML) koje nisu nove, ali se njihovim zajedničkim korištenjem unutar web preglednika može stvoriti bogatije korisničko okruženje [32]. Za prikaz sadržaja *Ajax* koristi standardni HTML i CSS. Dinamička promjena sadržaja na web stranici moguća je na dva načina. Prvi način je korištenjem objektnog modela HTML dokumenta (engl. *Document Object Model*, skraćeno DOM) koji definira dijelove HTML dokumenta. Korištenjem DOM-a moguće je promijeniti izgled ili sadržaj dijelova HTML dokumenta tijekom rada, bez potrebe za višekratnim učitavanjem čitave web stranice. Drugi, jednostavniji način je korištenje *IFrame* HTML elementa koji omogućuje uključivanje sadržaja druge web stranice unutar postojeće. Sadržaju umetnutom unutar *IFrame* objekta može se izravno pristupati i mijenjati njegov sadržaj. Za prijenos podataka između klijenta i poslužitelja koristi se HTTP protokol i tekstualni formati XML i JSON (*JavaScript Object Notation*). JSON je tekstualni format za razmjenu podataka, baziran na dijelu specifikacije standarda JS jezika i neovisan o tehnologiji [107]. Jedna od njegovih karakteristika je da se pomoću njega može preslikati struktura JS objekata čime se olakšava i pojednostavljuje prijenos podataka između klijenta i poslužitelja. Za asinkroni prijenos podataka koristi se *XMLHttpRequest* koji predstavlja dio API-ja za pristup funkcionalnostima HTTP protokola [124]. Sve prethodno navedene tehnologije povezuje skriptni jezik koji se izvršava unutar web preglednika. Izvorno je kao skriptni jezik korišten JS, no tijekom korištenja u praksi se pokazalo da se za tu namjenu može koristiti bilo koji skriptni jezik.

Najveći doprinos *Ajaxa* je promjena u načinu funkcioniranja web aplikacija sa sinkronog u asinkroni. Sadržaj se prikazuje pomoću HTML koda pa je jednostavno dostupan web pretraživačima. Osnovni nedostaci posljedica su načina na koji su aplikacije uklopljene u web preglednik. Različiti web preglednici ne interpretiraju HTML, CSS i JS kod na isti način pa nema garancije da će web stranica jednako izgledati u različitim web preglednicima. *Ajax* se bazira na skriptnim jezicima, pa ako korisnik u svom web pregledniku isključi podršku tim jezicima neće moći koristiti funkcionalnost web stranice. Unatoč navedenim problemima, većina Web 2.0 aplikacija koristi u većoj ili manjoj mjeri *Ajax* za oblikovanje klijentskog sučelja.

3.3.2.3 RIA tehnologije bazirane na vanjskim dodacima

Ova skupina tehnologija omogućuje bogatije korisničko sučelje i prikaz multimedijalnih podataka u odnosu na *Ajax*. Da bi se mogla koristiti njihova funkcionalnost, unutar web

preglednika se mora instalirati odgovarajući vanjski dodatak koji ovisi o konkretnoj tehnologiji. Korisnici su nepovjerljivi prema instalaciji takvih programa, osim prema zvučnim imenima proizvođača poput *Adobe*, *Microsofta* i sl. Problem kod korištenja takvih tehnologija je što se podaci unutar njih obično pohranjuju u binarnom formatu pa ih web pretraživači teško mogu indeksirati. Razlog je što većina RIA tehnologija nije otvorenog tipa već su u vlasništvu proizvođača. Zbog navedenih problema te tehnologije se rijetko koriste za izradu čitavih aplikacija koje su javno dostupne na webu, a puno češće za izradu aplikacija na *intranetu* ili za izradu dijelova aplikacija koji rade s multimedijalnim podacima. U posljednjih nekoliko godina uočljiv je snažan trend stvaranja i korištenja multimedijalnih podataka na webu [105], a aplikacije koje služe za pregledavanje tih sadržaja uglavnom su rađene u *Adobe Flashu*. To je jedan od razloga zbog čega je *Flash player* trenutno najpopularniji vanjski dodatak koji danas (srpanj 2009.) instaliran u svom web pregledniku ima preko 95% korisnika weba [102][111].

Povijesno, prva platforma za izradu RIA bili su *Java appleti* koji su se počeli koristiti 1995. godine, no oni nikada nisu stekli popularnost. Prva verzija *Flash* platforme proizvedena je 1996. godine, a veću popularnost je stekla s verzijom *Flash* MX 2002. godine. Osnovu platforme čini programski jezik *ActionScript* (skraćeno AS) razvijen na osnovu JS standarda. Zamišljena kao platforma za prikaz jednostavnih animacija i aplikacija, *Flash* platforma je prerasla u okruženje koje omogućuje izradu zahtjevnijih web aplikacija. Danas se *Flash* smatra neformalnim standardom za prikaz multimedijalnih podataka na webu jer većina web sjedišta za prikaz podataka te vrste koristi *Flash* aplikacije. Druge dvije popularne platforme koje su razvijene na istim postavkama i po svojoj funkcionalnosti slične *Flash* okruženju su *Microsoft Silverlight* [URI21] i *JavaFX* [URI19].

3.4 Utvrđivanje identiteta korisnika u Web 2.0 okruženju

U današnjim sustavima na webu upravljanje identitetom korisnika uglavnom je centralizirano što znači da svaki sustav samostalno vodi evidenciju o identitetu svojih korisnika. Takav pristup korisnicima stvara niz problema jer se prilikom prvog korištenja svakog web servisa moraju registrirati te izabrati i zapamtiti korisničko ime i lozinku. Nakon toga moraju pamtiti sva izabrana korisnička imena i lozinke. Autorima web aplikacija to donosi dodatne troškove u razvoju jer u njih moraju ugraditi jake sigurnosne mehanizme.

Alternativni pristup je decentralizirani način provjere kod kojeg je za provjeru identiteta korisnika zadužena treća strana kojoj vjeruju i korisnik i pružatelj usluga. Korisnik stvara svoj profil kod određenog davatelja (engl. *provider*) elektroničkog identiteta. Nakon što se identificira svome davatelju identiteta može koristiti različite usluge na nivou čitave mreže bez potrebe za ponovnom identifikacijom [22]. Takav identitet se obično naziva zajednički ili združeni identitet (engl. *federated identity*) i prikladan je za korištenje u okruženju u kojem korisnik koristi veći broj usluga različitih davatelja. Zbog svojih karakteristika, taj pristup se naziva jednostruka prijava (engl. *Single Sign On,* skraćeno SSO) [51]. Prilikom takvog načina provjere identiteta, korisnik ne mora pamtiti različita korisnička imena i lozinke već se prijavljuje na jednom mjestu i treba zapamtiti samo korisničko ime i lozinku toga mjesta.

3.4.1 Model decentraliziranog načina provjere identiteta

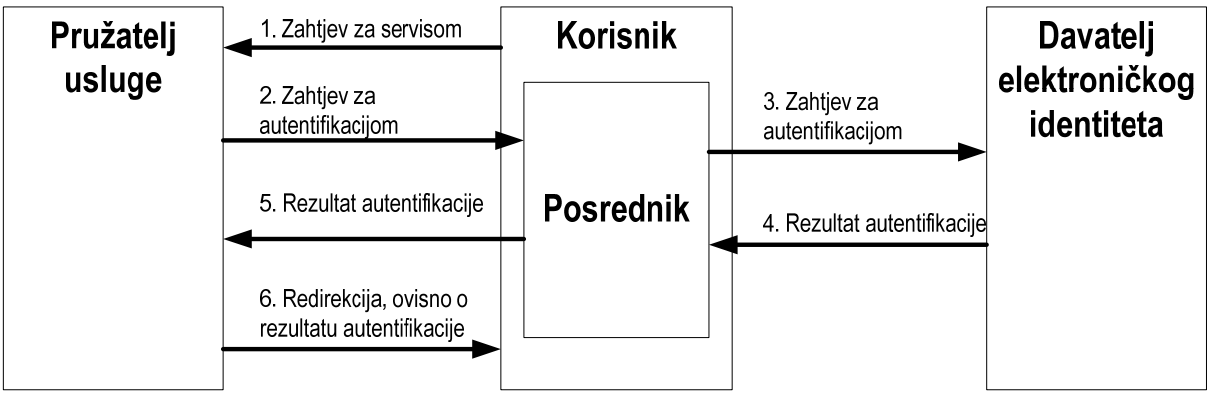
U SSO procesu javljaju se četiri različite uloge [51]:

- *Korisnik* je osoba koja ima digitalni identitet kod određenog davatelja elektroničkog identiteta i s tim identitetom se identificira prilikom korištenja različitih web aplikacija;
- *Posrednik u procesu provjere identiteta* (engl. *user agent*) je klijentska aplikacija zadužena za automatsko prosljeđivanje primljenih podataka te preusmjeravanje između web stranica pružatelja usluge i davatelja elektroničkog identiteta;
- *Pružatelj usluge* (engl. *service provider*) je web aplikacija čiju funkcionalnost korisnik želi koristiti i pri tome mora dokazati svoj identitet;
- *Davatelj elektroničkog identiteta* (engl. *identity provider*) je web aplikacija na koju se korisnik prijavljuje i koja pružatelju usluga dostavlja dokaz o njegovom identitetu.

Ovisno o načinu prijenosa podataka između dviju strana, razlikuju se dvije varijante utvrđivanja identiteta. Prva je pokrenuta od strane pružatelja usluga, a druga od strane davatelja elektroničkog identiteta.

Npr. korisnik pristupa web stranici investicijskog fonda na kojoj pregledava različite javno dostupne informacije. Ako želi pristupiti zaštićenom sadržaju, poput svog portfelja u investicijskom fondu, pružatelj usluge (u ovom slučaju to je aplikacija investicijskog fonda) mora poslati zahtjev za utvrđivanje identiteta korisnika njegovom davatelju elektroničkog identiteta (npr. njegovoj banci). U ovoj varijanti proces provjere identiteta je pokrenut od strane pružatelja usluga.

Na slici 3.9 prikazani su koraci provjere identiteta korisnika kada je proces pokrenuo pružatelj usluga.

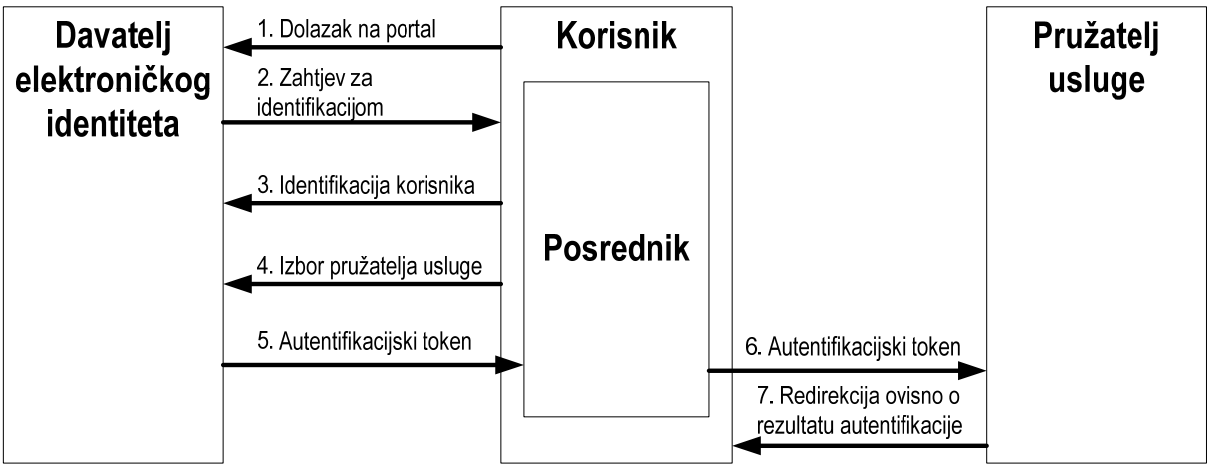


Slika 3.9 Proces provjere identiteta pokrenut od strane pružatelja usluga

Alternativa je proces pokrenut od strane davatelja elektroničkog identiteta. Kod tog modela, davatelj elektroničkog identiteta obavlja funkciju portala preko kojeg korisnik pristupa različitim zaštićenim servisima. Korisnik putem tog portala pristupa sadržajima različitih

pružatelja usluga. Npr. ima uvid u stanje svog bankovnog računa, mirovinskog fonda ili ocjena koje su njegova djeca dobila u školi.

Na slici 3.10 prikazani su koraci provjere identiteta korisnika u slučaju kada je inicijator tog procesa davatelj elektroničkog identiteta.



Slika 3.10 Proces provjere identiteta pokrenut od strane davatelja elektroničkog identiteta

Od opisanog modela provjere identiteta korist imaju sve strane. Korisnik se prijavljuje samo jednom, ne mora pamtiti mnoštvo različitih korisničkih imena i lozinki, a jednom prijavljen ima pristup uslugama raznih pružatelja usluga. Pružatelj usluga ne mora implementirati vlastiti sustav za provjeru identiteta čime ostvaruje uštedu i može se fokusirati na osnovne usluge, a davatelj elektroničkog identiteta se može posvetiti izradi što sigurnijeg sustava [51].

3.4.2 Potencijalni problemi u korištenju

Decentralizirani model provjere identiteta ima niz prednosti u odnosu na centralizirani, ali njegovo korištenje dovodi do potencijalnih problema vezanih uz sigurnost korištenja i privatnost korisnika. Podaci o identitetu korisnika razmjenjuju se između pružatelja usluge i davatelja elektroničkog identiteta koji se nalaze na različitim internetskim domenama. Sve strane u komunikaciji trebale bi osigurati svoje komunikacijske veze kako bi spriječili upade i krađu podataka, što mogu ostvariti korištenjem SSL protokola. Drugi sigurnosni problem proizlazi iz toga što se zaštita najčešće bazira na unosu kombinacije korisničkog imena i lozinke, što je dosta slaba zaštita. Pri prijenosu podaci se pohranjuju u obliku žetona (engl. *token*), a u sustavu u kojem se podaci razmjenjuju između poslužitelja na različitim internetskim domenama postoji velika opasnost od krađe tih podataka. Kako bi se ta opasnost umanjila *tokeni* obično imaju ograničen vijek trajanje. Ako npr. druga strana ne primi *token* u roku od jedne minute, izdani *token* postati će nevažeći. Kao dodatnu mjeru zaštite, neki od korištenih protokola omogućuju automatsku odjavu korisnika. Kada se korisnik odjavi sa stranice jednog davatelja usluga, sustav će ga automatski odjaviti sa stranica svih davatelja usluga kod kojih je prijavljen [51].

Pošto se jedan identitet koristi za pristup većem broju aplikacija, postoji opasnost od narušavanja privatnosti korisnika uslijed praćenja njegovih mrežnih aktivnosti. Jedan od

načina zaštite je korištenje pseudonima prilikom stvaranja digitalnog identiteta. Npr. korisnik se prijavljuje s pseudonimom Pero. Sustavi koji komuniciraju znaju tko je Pero, međutim napadač koji prati aktivnosti teško može povezati taj pseudonim s podacima konkretne osobe. Pored toga, korisnik može kreirati više različitih digitalnih identiteta i koristiti ih za različite namjene.

3.4.3 Protokoli

Tri trenutno najpopularnija protokola za distribuirano upravljanje identitetom korisnika su SAML (*Security Assertion Markup Language*) [112], *OpenID* [109] i *InfoCard* [58]. SAML protokol je najstariji, baziran je na XML-u i orijentiran korištenju u poslovnim okruženjima, naročito u kombinaciji sa SOAP protokolom. *InfoCard* je *Microsoft* .NET komponenta razvijena za potrebe upravljanja digitalnim identitetom korisnika u *Microsoft* okruženju. *OpenID* protokol razvijen je 2005. godine za provjeru identiteta autora sadržaja na webu kako bi se smanjila količina neželjenih poruka u sustavu. Osnovna želja autora je bila da bude što jednostavniji, a autori su ga razvijali imajući na umu filozofiju osobnog korisničkog identiteta. Korisnici mogu vrlo jednostavno postati davatelji elektroničkog identiteta, pa zbog toga protokol postaje sve popularniji na webu, a mnoge web aplikacije omogućuju svojim korisnicima *OpenID* identitet. Početkom srpnja 2009. godine broj web sjedišta koja podržavaju provjeru identiteta korisnika korištenjem *OpenID* protokola je oko pedeset tisuća, a u posljednjih sedam mjeseci taj broj je porastao za dvadeset tisuća [110].

3.5 Web 2.0 aplikacije

3.5.1 Elementi Web 2.0 aplikacija

Svaka uspješna Web 2.0 aplikacija sadrži određene funkcionalnosti [54]:

- jednostavno korisničko sučelje;
- kvalitetan pretraživač sadržaja;
- povezivanje sadržaja;
- mogućnost označavanja podataka u sustavu korisničkim oznakama;
- mogućnost jednostavnog objavljivanja sadržaja;
- pomoć korisnicima predlaganjem zanimljivih sadržaja ili aktivnosti;
- praćenje novosti.

Sučelje bi trebalo biti što je moguće jednostavnije, a korisnik bi u svakom trenutku u okruženju aplikacije trebao lako pronaći traženu informaciju na isti način kako to radi na webu korištenjem klasičnih web pretraživača. RIA-e bazirane na *Ajax* tehnologiji sadržaje prikazuju u HTML obliku i time omogućuju korištenje klasičnih web pretraživača. Vrijednost nekog sadržaja na webu može se dosta dobro procijeniti na osnovu toga koliko se drugi korisnici vežu na njega. Pretraživač *Google* [URI13] je koristeći tu spoznaju napravio

revoluciju u pretraživanju web sadržaja uključivši u algoritam rangiranja web stranica informaciju o broju veza prema toj stranici.

Osim kvalitetnog pretraživača, korisnici žele mogućnost pristupa materijalima putem odgovarajuće klasifikacijske strukture. U suradničkom okruženju teško je očekivati da će autori slijediti formalno zadanu strukturu, pa je najbolje što se može učiniti omogućiti im da materijale označavaju slobodno prema vlastitom nahođenju i kreiraju folksonomiju koja se može koristiti prilikom pretraživanja.

Korisnici weba vole stvarati i objavljivati materijale na webu [105] pa im to treba i omogućiti. Alati za stvaranje i objavu sadržaja trebaju biti što je moguće jednostavniji jer većina korisnika weba nisu informatički stručnjaci, a kreirani sadržaj treba biti u standardnom HTML formatu kako bi bio čitljiv različitim web preglednicima. Na osnovu poznavanja korisnikovih navika (npr. na osnovu analize uzoraka korisničkih oznaka) sustav bi trebao imati funkcionalnost predlaganja sadržaja ili aktivnosti koje su u skladu s korisnikovim interesima. Npr. tvrtka *Amazon* [URI2] na osnovu analize kupovina, određenom kupcu nudi informacije o proizvodima koji bi ga potencijalno mogli zanimati. Pri tome treba imati mjeru jer bi preosobne preporuke kod korisnika mogle stvoriti odbojan stav u stilu „veliki brat me gleda“.

U današnje vrijeme na webu se svakodnevno stvara ogromna količina materijala i teško je svakodnevno pratiti sve novosti. Kao pomoć u tome dolaze tehnologije koje omogućuju pretplatu na željene sadržaje koje korisniku isporučuju novostvorene materijale. Primjer takvih tehnologija su RSS *feedovi* ili *Google Alerts* [URI14] koji omogućuju redovnu isporuku sadržaja na koje su korisnici pretplaćeni. Kod RSS *feedova*, korisnik informacijama pristupa putem čitača RSS sadržaja, a kod korištenja *Google Alerts* usluge, korisniku na prijavljenu adresu elektroničke pošte stižu informacije o novim web stranicama koje je indeksirao web pretraživač *Google*.

3.5.2 Primjeri Web 2.0 aplikacija

Brojne Web 2.0 aplikacije i tehnologije mogu se uspješno koristiti prilikom samostalnog učenja, a neke od njih koriste se i u suvremenim VLE-ovima. U nastavku će se opisati neke od takvih aplikacija.

3.5.2.1 Blog

Blog je web aplikacija u kojoj vlasnik objavljuje tekstualne materijale u obliku dnevnika u kojem se objavljeni materijali prikazuju poredani u obratnom kronološkom redu (prvo se prikazuju posljednji napisani članci, pa sve stariji). Blogovi su se počeli koristiti sredinom devedesetih godina 20. stoljeća i mogu se smatrati najstarijom Web 2.0 aplikacijom [118]. Prvi blogovi su bili jednostavne web aplikacije koje su se bazirale na objavi podataka na statičnim web stranicama. S vremenom je popularnost samoizdavaštva porasla, mogućnosti aplikacija su se povećale, a korištenje pojednostavnilo. Unatoč tim promjenama osnovne osobine i namjena su ostali isti. Čitatelji blogova imaju mogućnost komentiranja članaka i davanja povratnih informacija autoru. Osobine blogova su:

- jednostavnost stvaranja – svatko bez imalo znanja o web programiranju može samostalno stvoriti blog te objavljivati sadržaje;
- lakoća pristupa – korištenjem web preglednika ili RSS čitača moguće je pristupati novim sadržajima objavljenima na blogu;
- preglednost – sadržaji su kronološki poredani te pružaju dodatne informacije kao npr. tko je objavio sadržaj, tko je sve dao komentar i sl.

Blogovi su danas uvjerljivo najposjećeniji sadržaji na webu i važan medij za objavu novih informacija [105]. Posebna vrsta blogova su mikroblogovi koji su prilagođeni korisnicima koji vole kraće sadržaje u skladu sa suvremenim ubrzanim tempom života. Razvijeni su po uzoru na uspješnu upotrebu SMS-ova i MMS-ova u mobilnoj telefoniji. Trenutno najpopularniji servis te vrste je *Twitter* [URI36] koji omogućuje objavu i praćenje kratkih poruka duljine do 140 znakova.

3.5.2.2 Wiki

Wiki sustavi su repozitoriji informacija u obliku web stranica koje korisnici mogu jednostavno uređivati i na njima objavljivati povezane informacije. Razvijeni su u isto vrijeme kao i blogovi. Za razliku od blogova, wiki sustavi nisu ograničeni na kronološko objavljivanje informacija. U wiki sustavu više korisnika ravnopravno uređuje sadržaje pomoću posebnog jednostavnog jezika. U wiki sustavu više korisnika ravnopravno sudjeluje u procesu stvaranja novih informacija. Kako korisnici ne bi morali učiti HTML većina wiki sustava uvela je i grafičke uređivače sadržaja [123].

Pristup zajedničkom uređivanju sadržaja od strane više korisnika na početku se činio nemogućim i vrlo riskantnim zbog slučajnog ili namjernog objavljivanja pogrešnih informacija. *Wikipedia* [URI38] je dokazala da takav pristup upravljanja informacijskim resursima unatoč povremenim problemima može rezultirati točnim i stabilnim sadržajem [46][43]. Bez obzira na nesuglasice i probleme koji se povremeno javljaju, prema istraživanju koje je prije nekoliko godina proveo časopis *Nature*, kvaliteta i točnost informacija o znanstvenim temama na *Wikipedijinim* stranicama usporediva je s kvalitetom i točnošću informacija *Enciklopedije Britannice* [33]. Unatoč nekim prigovorima na metodologiju i način na koji je provedeno istraživanje [101], može se smatrati da su informacije na Wikipediji točne. Članke na Wikipediji stvaraju volonteri što je princip na kojem se baziraju sva suradnička okruženja na webu. Što više osoba promatra neki članak, prije će uočiti netočnosti u njemu, a u slučaju spora konačnu odluku o tome što će u članku biti objavljeno imaju urednici odgovarajućih područja. Naravno, pri korištenju informacija s Wikipedije, kao i ostalih sličnih izvora, uvijek je informacije potrebno uzeti s dozom rezerve i obavezno ih provjeriti iz drugih relevantnih izvora.

3.5.2.3 Servisi za pohranu knjižnih oznaka

Današnji web je gotovo neograničen izvor informacija. Početkom kolovoza 2009. godine na webu se nalazilo oko 48 milijardi web stranica [116]. Velik problem za korisnike je

pronalaženje traženih informacija u toj masi. Na webu je svaki sadržaj dostupan putem svog URI-ja. Da bi se sadržaj pohranio za ponovni dohvat, dovoljno je pohraniti njegov URI, i po želji dodatne podatke koji mogu pomoći pri pronalaženju sadržaja (npr. naslov, opis i korisničke oznake).

Pohranjivanje knjižnih oznaka u okruženju društvene mreže (engl. *social bookmarking*) obuhvaća pohranjivanje, klasificiranje, i označavanje materijala na webu kako bi ti sadržaji naknadno bili lakše dostupni i pretraživi, ne samo autorima već i ostalim korisnicima [122]. Mogućnosti pohrane knjižnih oznaka korisnici weba su koristili od samih početaka. Svaki web preglednik ima mogućnost pohrane knjižnih oznaka pa su ih na početku korisnici pohranjivali unutar web preglednika. Problem nastaje jer se podaci pohranjuju lokalno na računalo na kojem je instaliran web preglednik. Kada neka osoba koristi više različitih računala ne može pristupati svim zabilježenim podacima sa svakog računala. Rješenje je da se knjižne oznake pohranjuju na mreži.

Web 2.0 servisi za pohranu knjižnih oznaka popularnost su stekli zahvaljujući jednostavnosti korištenja koju je omogućilo *Ajax* sučelje i širokoj bazi korisnika koji su imali mogućnost ne samo zabilježiti adresu odgovarajućih sadržaja već ih označavati vlastitim oznakama. Prava snaga tih servisa je u tome što za svaki zapisani URI daju informaciju koliko je drugih korisnika zabilježilo neki sadržaj, te kojim su ga oznakama označili što korisniku može biti dobar pokazatelj njegove kvalitete.

3.5.2.4 Servisi za pohranu dokumenata

Internetski servisi omogućavaju pohranu različitih vrsti podataka u elektroničkom obliku. Njihova osnovna prednost je dostupnost usluge s bilo koje lokacije s koje se moguće spojiti na Internet, te što prostor za pohranu dokumenata osigurava davatelj usluge. Prvi servisi za pohranu dokumenata na mreži služili su za sigurnosnu pohranu dokumenata. Korisnik bi pohranio dokumente na web poslužitelj gdje su bili zaštićeni od potencijalnih oštećenja uslijed kvarova računalne opreme i medija za pohranu. Web 2.0 servisi su promijenili osnovnu namjenu takvih servisa i ona više nije prvenstveno sigurnosna pohrana, već način da se materijali učine lako dostupnima. Jedan od prvih Web 2.0 servisa, popularni *Flickr* je u svojoj osnovi servis za pohranu slika. Današnji servisi za pohranu dokumenata su društvena okruženja u kojima korisnici stvaraju i pohranjuju dokumente koristeći diskovni prostor i računalne resurse davatelja usluge. Oni su specijalizirani po vrsti sadržaja koje pohranjuju. Tako npr. *YouTube* i *BlipTV* [URI5] služe za pohranu video materijala, *Docstoc* [URI9], *Scribd* [URI31] i *SlideShare* [URI34] za pohranu raznih dokumenata i prezentacija, a *Flickr* i *Picasa* [URI28] za pohranu slika. Postoje i servisi poput *Boxa* [URI7] i *Wualae* [URI41] koje služe za pohranu dokumenata bilo koje vrste. Kao i u većini ostalih društvenih mreža članovi imaju mogućnost pristupa, dijeljenja, komentiranja i označavanja svih materijala koje je autor označio javno dostupnima.

Negativna strana korištenja takvih servisa leži u tome što su podaci pohranjeni na tuđim poslužiteljima i autor nema onolike ovlasti nad dokumentima kolike bi imao kada bi ih pohranio unutar vlastitog okruženja.

4 OSOBNI OKOLIŠ ZA UČENJE

Osobni okoliš za učenje je svaki okoliš koji osoba koristi za samostalno upravljanje procesom vlastitog učenja. U ovom poglavlju opisati će se navedeni koncept i usporediti nekoliko načina za stvaranje takvog okoliša.

4.1 Potreba za razvojem osobnog okoliša za učenje

Danas se količina informacija i dostupnog znanja vrlo brzo povećava tako da su često informacije objavljene u klasičnim medijima poput knjiga i časopisa već u trenutku objave zastarjele zbog vremena koje protekne u procesu uređivanja i izdavanja [42]. Procjenjuje se da je vrijeme poluraspada informacija baziranih na razvoju znanosti i tehnike i tehnologije oko 5 godina, što znači da je stečeno znanje na tim područjima nakon 10 godina gotovo beskorisno. Posljedica toga je da je znanje potrebno neprekidno obnavljati i nadopunjavati nakon završetka formalnog obrazovanja što proces cjeloživotnog učenja čini neophodnim. Proces učenja je uglavnom upravljan od strane samih učenika, a pošto se najažurnije informacije nalaze na Internetu, on je postao nezaobilazan resurs.

4.1.1 Odmak od tradicionalnog modela učenja

Za vrijeme formalnog i neformalnog obrazovanja, učenik sudjeluje u procesu učenja kao dio zajednice. U takvom okruženju ima mogućnost komunikacije i razmjene informacija s nastavnicima i drugim učenicima. Kada učenik uči samostalno, on upravlja tijekom učenja bez nadzora mentora. Klasični VLE-ovi mogu odgovarati učenicima i studentima tijekom formalnog obrazovanja u obrazovnim institucijama, ali u slučaju informalnog učenja kod kojeg učenik samostalno određuje plan i bira materijale, njihov model ne odgovara. Proces učenja unutar VLE-a najčešće je jednoobrazan, putanja učenja usmjerena na način kako je to zamislio instruktor, a da bi došao do cilja učenik mora proći kroz sve teme zadanim redoslijedom. Materijali unutar VLE-a najčešće su zaključani i dostupni samo učenicima prijavljenima na određeni kolegij ili tečaj, a obično su im dostupni samo za vrijeme tečaja. To je u suprotnosti s osnovnim postavkama cjeloživotnog obrazovanja prema kojima je učenje trajan proces, a obrazovni materijali bi trebali biti dostupni učeniku za vrijeme čitavog života. Svaka osoba ima svoj vlastiti stil učenja i u nekim slučajevima stil nametnut kroz VLE im ne odgovara [92]. Zbog toga je vidljiv novi trend u učenju baziran na korištenju Web 2.0 alata koji donosi promjene u načinu stjecanja znanja i razmišljanja osoba koje se uglavnom obrazuju samostalno. Klasični model institucionalnog prijenosa znanja ih ne zadovoljava pa traže okruženje koje će im omogućiti veću slobodu pri izboru načina učenja, materijala, komunikacije i suradnje ne samo sa učenicima na istom tečaju već sa svim korisnicima weba koje zanima isto područje [23].

4.1.2 Konektivizam – teorija učenja u digitalnom dobu

Uočene promjene u načinu stjecanja znanja u suvremenom društvu potakle su teoretičare na otkrivanje postavki koje bi mogle opisati model učenja u novostvorenom okruženju. Znanje

koje su u prošlosti stjecale osobe tijekom svog formalnog obrazovanja trajalo bi najčešće do kraja njihovog radnog vijeka. Zbog toga je bilo važno pronaći optimalne metode stjecanja znanja koje će omogućiti učinkovitu pohranu činjenica i njihovo korištenje tijekom života. Klasične teorije učenja fokusirane su na takav pristup učenju. Razvoj tehnologije utjecao je na promjene u strukturi znanja, tako da se u mnogim područjima upotrebljivost postojećeg znanja smanjila. Pohrana činjenica više nije toliko važna jer se one brzo mijenjaju, a mnogo važnije postaje znanje o tome kako neprekidno usvajati novo znanje i kritički preispitivati postojeće.

Konektivizam je teorija učenja u digitalnom dobu razvijena na osnovnoj postavci da osobe moraju neprekidno tragati za novim informacijama. U suvremenom okruženju učenje nije isključivo individualna aktivnost već je u tom procesu važna povezanost s drugim osobama, a u tome značajno pomažu tehnološka dostignuća i Web 2.0 okruženja. Osnovna vještina koju učenici moraju imati je sposobnost kritičkog preispitivanja i vrednovanja informacije. Mreža veza s drugim učenicima koju stvaraju tijekom učenja mnogo je važnija od konkretnog znanja koje u određenom trenutku postoji u toj mreži. Uzrok tome je činjenica što se količina postojećeg znanja neprekidno povećava, a ključne postaju sposobnost prepoznavanja mjesta na koje treba uklopiti nove informacije i mogućnost provjere stečenog znanja u praksi [72].

4.1.3 Utjecaj tehnoloških promjena

Intenzivan razvoj i korištenje računalnih i komunikacijskih tehnologija uveo je mnoge promjene u način učenja iako se sam način prikupljanja, evidencije i stvaranja materijala nije mnogo promijenio u odnosu na nekadašnji. I dalje pronalazimo materijale polazeći iz kataloga ili na osnovu preporuka drugih osoba koje su koristile iste materijale. Ono što se znatno promijenilo je oblik, dostupnost i brzina pronalaženja materijala.

Prije dvadesetak godina učenik u malom mjestu bez dobro opremljene biblioteke je imao ogromnih problema u pronalaženju literature, posebno stručne. To je za njega bio težak zadatak jer bi se morao zaputiti na put u veći grad koji ima znanstvenu biblioteku, ili bi morao naručivati i kupovati knjige. Sam proces isporuke iz inozemstva trajao je tjednima, a ponekad i mjesecima. Većina materijala bila je dostupna samo u papirnatom obliku. Danas je mnoštvo informacija dostupno na webu u elektroničkom obliku, a u pristupu tim informacijama jedino ograničenje je postojanje računala i kvalitetne veze na Internet. Oblik u kojem se objavljuju sadržaji se promijenio pa je sve više materijala dobavljivo u elektroničkom obliku, posebno u obliku zvučnih ili video zapisa.

Druga stvar koja se promijenila je način komunikacije s drugim osobama. Nekada je ta komunikacija bila isključivo osobna, putem telefona ili pisama. Tehnologija na webu, a posebno Web 2.0 usluge koje se baziraju na korištenju društvenih mreža su to promijenile pa danas svatko, bez obzira na mjesto u kojem živi može vrlo jednostavno komunicirati putem Interneta s drugom osobom koja živi na drugom kraju svijeta.

4.2 Koncept osobnog okoliša za učenje

PLE nije konkretna tehnologija ili aplikacija već je to koncept korištenja različitih tehnologija i aplikacija u procesu samostalnog učenja. Ključna odrednica koncepta je da korisnik samostalno određuje koje će aplikacije koristiti i za koju namjenu.

Danas je na webu dostupno mnoštvo materijala i aplikacija koje se mogu učinkovito upotrijebiti u samostalnom učenju. Da bi se oni mogli učinkovitije koristiti i da bi im se dodala nova vrijednost, potrebno ih je na odgovarajući način klasificirati, urediti i povezati u cjelinu. Novi način cjeloživotnog učenja zahtjeva okruženje koje će učenicima omogućiti prikupljanje, označavanje, uređivanje i povezivanje resursa koje će koristiti tijekom učenja, te njihovu pohranu tako da budu trajno dostupni. Takvo okruženje mora biti jednostavno za održavanje i administraciju i ne smije zahtijevati specifično tehničko predznanje korisnika [92].

4.2.1 Definicija pojma

Ideja PLE-a nastala je kao rezultat razmišljanja o tome na koji način prilagoditi računalnu podršku procesima cjeloživotnog informalnog učenja. Tehnička realizacija okruženja uglavnom je inspirirana popularnošću novih Web 2.0 koncepata i aplikacija. Jedinstvena definicija PLE-a ne postoji, a svaki autor naglašava određeni segment takvog okruženja. Jedna od općenitih definicija kaže da je PLE svako okruženje pomoću kojeg pojedinac može prikupljati i koristiti materijala u digitalnom obliku tijekom svog procesa učenja [50]. Prema Van Harmelenu [81] osobni okoliš za učenje je sustav koji omogućuje korisniku preuzimanje potpune kontrole i upravljanje vlastitim učenjem. To obuhvaća podršku za stvaranje vlastitih obrazovnih ciljeva, upravljanje samim procesom učenja, prikupljanje materijala te komunikaciju s drugim sudionicima koji uče o istoj temi. M. A. Chatti [14] smatra da su PLE-ovi karakterizirani korištenjem različitih servisa i alata tijekom učenja nad kojima učenik ima potpunu kontrolu. Prema S. Downesu [21], PLE je bilo koje okruženje koje omogućuje zadovoljavanje tri principa: interakciju, korisnost i relevantnost. Interakcija obuhvaća komunikaciju s drugim sudionicima tijekom procesa učenja korištenjem dostupnih alata. Za vrijeme učenja stvaraju se interesne mreže u kojima učenici međusobno komuniciraju, razmjenjuju informacije i materijale. Informacije čine ključni resurs i učenik bi trebao dobiti relevantne informacije onda kada su mu potrebne, u obliku koji mu u tom trenutku najbolje odgovara.

4.2.2 Osnovne karakteristike osobnog okoliša za učenje

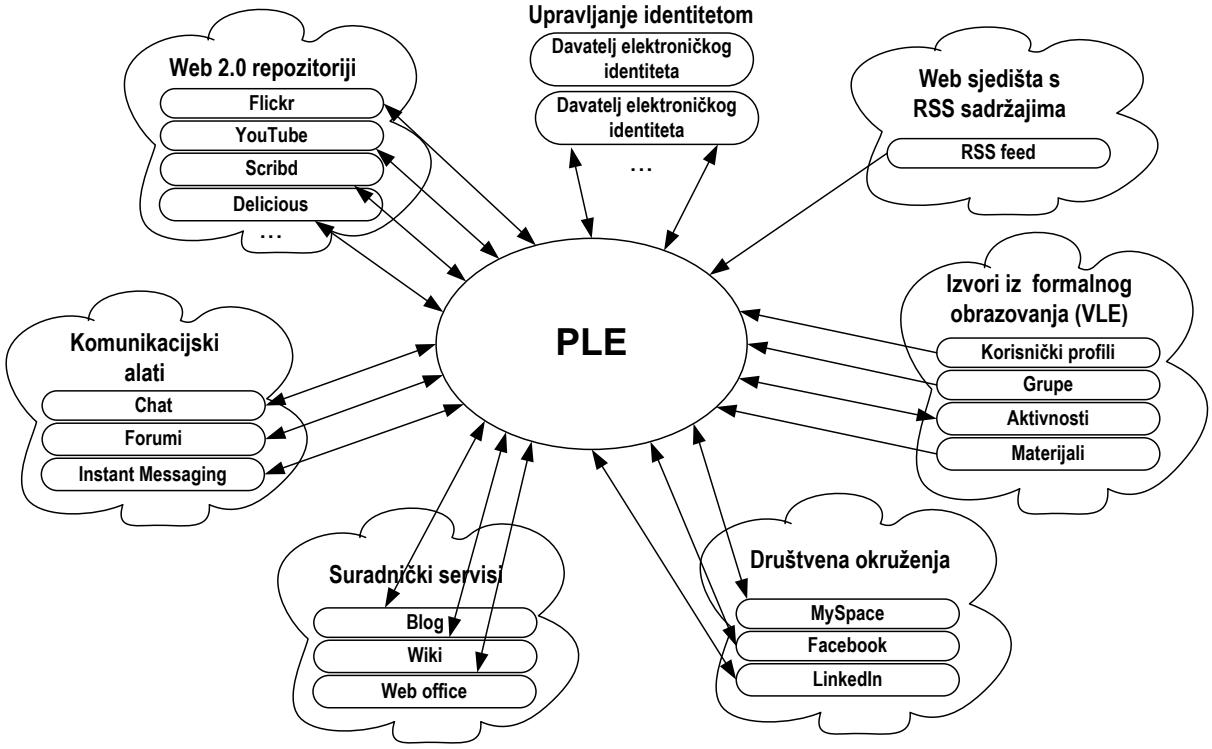
Bez obzira na širok pregled definicija i različite interpretacije termina, mogu se izdvojiti četiri karakteristike koje bi PLE trebao zadovoljavati [56]:

1. Uspostavljanje veza s drugim učenicima. Okruženje treba omogućiti uspostavljanje neformalnih veza među učenicima i njihovu međusobnu komunikaciju i suradnju.
2. Upravljanje obrazovnim resursima. Okruženje treba omogućiti uređivanje, strukturiranje, označavanje i razmjenu materijala koje je učenik kreirao, te pronalaženje, označavanje i korištenje materijala iz drugih izvora.
3. Upravljanje aktivnostima. Okruženje treba pružiti podršku odvijanju pojedinačnih i grupnih aktivnosti učenika poput izrade plana učenja, prikupljanja i komentiranja materijala ili grupnog rada.
4. Integracija s okruženjem. PLE bi se trebao moći jednostavno povezati s drugim okruženjima koja se koriste u procesu učenja (npr. s VLE-om obrazovne institucije). Najjednostavniji oblik takvog povezivanje je uključivanje materijala iz drugih okruženja u PLE, a složeniji oblici uključuju korištenje funkcionalnosti vanjskih okruženja unutar PLE-a, npr. u obliku *widgeta* [86].

Sadržaj PLE-a je kolekcija različitih alata i resursa dostupnih na webu koje osoba organizira prema svojim željama i potrebama. Kod stvaranja takvog okruženja važno je voditi računa o činjenici da većina korisnika Interneta nisu informatički stručnjaci. Mnoštvo dostupnih materijala i alata često će zbuniti i vješte korisnike weba. Za njihovo učinkovito korištenje potrebno je sučelje koje će pomoći u organizaciji sadržaja i pristupa servisima koji se koriste. To sučelje bi trebalo omogućiti pristup sadržajima i aktivnostima, njihovo povezivanje u smislenu strukturu te stvaranje, označavanje, pretraživanje i pohranjivanje materijala. Ono služi kao most između korisnika i alata koji se nalaze na webu, a od korisnika bi trebalo skrivati sve nepotrebne detalje implementacije. To sučelje predstavlja računalnu podršku stvaranju osobnog okoliša za učenje.

4.2.3 Konceptualni model osobnog okoliša za učenje

Na slici 4.1 prikazan je konceptualni model PLE-a baziran na modelu virtualnog okruženja budućnosti koje je 2005. godine predložio Scott Wilson [87].



Slika 4.1 Konceptualni model PLE-a

PLE je agregator koji služi za povezivanje svih materijala i aktivnosti koje osoba koristi u učenju tijekom svog života. U takvom okruženju važno je upravljanje elektroničkim identitetom korisnika pošto on koristi različite servise i resurse. Ako se osoba obrazuje formalno ili neformalno, dio aktivnosti obavlja u okviru VLE-ova obrazovnih institucija. Znanje koje stvori u tom okruženju, kao i veze s drugim učenicima mogu biti koristan element unutar PLE-a. Problem je što su takvi materijali dostupni samo dok traje obrazovanje na određenoj instituciji. Osim toga većina VLE-ova su zatvoreni sustavi kojima nije moguć pristup od strane vanjskih sustava. Otvoreni sustavi omogućuju povezivanje, i na tom području se dosta napreduje, naročito unutar otvorenih VLE-ova poput *Moodlea*. Međutim veći problem je organizacijske prirode jer obrazovne institucije nerado dopuštaju ulaz takve vrste u svoj sustav.

Korištenjem web servisa mogu se pohranjivati materijali različitih vrsta, a komunikacija se može obavljati korištenjem klasičnih komunikacijskih alata (foruma, komunikaciju čavrljanjem, slanjem poruka) ili danas popularnijih društvenih aplikacija poput *MySpacea* ili *Facebooka*.

4.2.4 Usporedba virtualnog obrazovnog okruženja i osobnog okoliša za učenje

VLE-ovi su danas dominantan oblik sustava za računalom podržano učenje. Njihova funkcionalnost razvijena je ciljano kako bi se omogućio jednostavan prijenos znanja unutar obrazovnih institucija. Osobni okoliš za učenje s druge strane je potpuno osoban, a koncept je razvijen kao podrška promjeni dominantnog načina obrazovanja s formalnog na informalno [88]. Suvremeni VLE-ovi pokušavaju poboljšati način učenja tako što pripremljene materijale prezentiraju učeniku na način koji mu po njegovim osobinama najbolje odgovara. Modeli koji se izrađuju unutar takvog okruženja pokušavaju korištenjem statističkih metoda i metoda umjetne inteligencije odrediti osobine učenika kako bi mu prenijeli znanje na optimalan način. Osobni okoliš za učenje ne koristi nikakve posebne metode za usvajanje znanja, a njegova osnovna karakteristika je da je osoban i da ga učenik oblikuje prema svojim željama i potrebama.

Jay Cross je na svom blogu [18] zanimljivom metaforom opisao razliku između VLE-a i PLE-a. VLE je poput švicarskog vojnog nožića koji sadrži mnoštvo korisnih alata koji mogu pomoći kada je nešto potrebno na brzinu popraviti, a pri ruci vam nije pravi alat. PLE s druge strane predstavlja kutiju punu alata iz koje će svatko moći izabrati onaj alat koji mu je u određenoj prilici potreban.

Da bi osoba uspješno koristila PLE moraju biti zadovoljene dvije pretpostavke: osoba mora biti motivirana i mora znati samostalno učiti. Najbolja motivacija za većinu osoba je saznanje da će stečeno znanje pokazati u praksi. Motivirani učenik će se potruditi i samostalno oblikovati okruženje onako kako mu najbolje odgovara. Pri tome se može dogoditi slučaj da ode u pogrešnom smjeru i zbog toga je poželjno da u okruženju postoji neki korektiv poput drugih osoba koje poznaju tu materiju. Važna uloga formalnog obrazovanja trebala bi biti da polaznike nauči kako samostalno učiti. Oni koji tijekom formalnog školovanja steknu to znanje biti će osposobljeni za samostalno učenje u nastavku života, a drugi će imati problema.

U okruženju VLE-ova veze između sudionika su asimetrične. Nastavnik prenosi informacije učenicima, priprema materijale i time određuje tijek i način na koji učenici trebaju prolaziti kroz lekcije. Učenici su pasivni primatelji sadržaja koji imaju ograničenu mogućnost stvaranja sadržaja i donošenja odluke o načinu učenja. Unutar PLE-a veze su simetrične, a svi sudionici ravnopravno komuniciraju. Svaki učenik se može naći u ulozi nastavnika kada pomaže u usvajanju određenih sadržaja drugim učenicima.

VLE okruženje je identično za sve učenike, a nastavni plan je zajednički. Tijek učenja je organiziran tako da svi učenici koriste iste obrazovne materijale i alate i slijede plan. Dizajn okruženja bazira se na grupiranju obrazovnih sadržaja i njihovom oblikovanju u zaokružene cjeline unutar tečaja ili kolegija. Nastavnici i autori materijala time posredno utječu na uspjeh učenja jer će pristup koji oni izaberu jednim učenicima više odgovarati, a drugima manje. Prilagodljivi i inteligentni sustavi za podršku učenju nastoje ukloniti taj nedostatak prilagođavajući način prijenosa znanja osobinama učenika, međutim i u njima se u središtu nalazi nastavnik, a učenik nema mogućnost većeg utjecaja na oblik i vrstu nastavnih

materijala. PLE-ovi su orijentirani stvaranju veza s drugim učenicima i prikupljanju obrazovnih materijala. Unutar PLE-a korisnik samostalno odlučuje o vrsti materijala, aktivnostima i tijeku učenja, a prikupljeni resursi i ostvarene veze nisu ograničeni samo na učenje o jednoj temi.

Sadržaj unutar VLE-a uglavnom je ograničen na polaznike koji pohađaju određeni kolegij ili tečaj. Po završetku nastave učenici nemaju pristup sadržajima koje su koristili i stvorili unutar sustava. Takve karakteristike ne zadovoljavaju potrebe cjeloživotnog obrazovanja u kojem bi osoba trebala imati pristup obrazovnim materijalima cijelog svog života. Za razliku od VLE-ova, u PLE-ovima sami učenici odlučuju koliko dugo će čuvati određeni sadržaj.

Za potrebe korištenja VLE-ova razvijeni su razni standardi koji se uglavnom odnose na vezu VLE-ova s drugim sustavima i razmjenu obrazovnih materijala. PLE-ovi su smješteni van obrazovnih institucija na webu. Za njihovo funkcioniranje i komunikaciju s drugim sustavima na webu nužno je korištenje otvorenih internetskih standarda (npr. RSS, RDF ili *Atom* za objavu sadržaja, XML-RPC, SOAP ili RESTful web servisi za korištenje funkcionalnosti drugih web aplikacija).

VLE-ovi su institucionalno orijentirani i razvijeni kao podrška modelu u kojem manji broj predavača prenosi znanje većem broju učenika. Zbog toga se u njihovom središtu nalazi institucija, a velika pažnja se posvećuje administrativnom dijelu sustava. U središtu PLE-ova nalazi se učenik koji je stvorio i oblikovao okruženje prema svojim željama i potrebama, a korištenje i administracija trebaju biti što je moguće jednostavniji.

4.3 Povijesni razvoj osobnih okoliša za učenje

4.3.1 Nastanak ideje

1998. istraživačka grupa sa Sveučilišta u Helsinkiju osmislila je obrazovno okruženje budućnosti (engl. *Future Learning Environment*) koje je za to vrijeme bilo revolucionarno pošto je uključivalo koncepte učenja u suradničkom okruženju unutar kojega su učenici i mentori ravnopravno razmjenjivali informacije i stvarali novo znanje. Po završetku tečaja stvoreno znanje bi ostajalo u sustavu i bilo dostupno drugim učenicima u budućnosti [78].

Colloquia je sustav za grupno učenje i rad u okruženju ravnopravnih sudionika (engl. *peer-to-peer*). Korisnici se unutar sustava okupljaju oko zajedničke teme i imaju mogućnost prikupljanja, pohrane i razmjene materijala s drugim sudionicima. Komunikacijska komponenta sustava je vrlo važna i korisnici međusobno mogu komunicirati tekstualnim porukama. Prva verzija sustava razvijena je 2000. godine, a sam sustav je tehnološki bio realiziran kao okruženje ravnopravnih sudionika [49].

Na Sveučilištu u Manchesteru, 2004. godine završen je projekt izrade prve verzije sustava za udaljeno učenje koji je imao za cilj izradu programske podrške koju će studenti sveučilišta moći koristiti pored VLE-a i u koji će moći uključivati osobne materijale. Programska podrška bila je napravljena u obliku klijentsko-poslužiteljske aplikacije [79].

Tijekom 2005. godine na sedam sveučilišta u SAD-u provedeno je istraživanje VLE-ova intervjuiranjem svih skupina njihovih korisnika: učenika, nastavnika i administrativnog osoblja. Na osnovu rezultata [40] zaključeno je da je potrebna promjena modela sustava za računalom podržano učenje kako bi bili prilagođeniji učenicima i podržali zahtjeve samostalnog cjeloživotnog učenja. U takvom sustavu VLE predstavlja samo jedan mali segment koji podržava formalni dio obrazovanja.

4.3.2 Utjecaj društvenih mreža na razvoj osobnih okoliša za učenje

Na daljnji razvoj računalnih sustava za stvaranje PLE-ova utjecala je popularnost Web 2.0 koncepta i aplikacija koje omogućavaju rad u društvenim mrežama. 2004. godine kompanija *Robot Coop* izradila je društvenu mrežu *43Things* [URI1] u kojoj su članovi mogu navoditi osobne ciljeve te komunicirati i surađivati s drugima na ostvarenju zacrtanih ciljeva. Brzo je zapaženo da je velik broj korisnika postavljao osobne obrazovne ciljeve. Dizajn tog okruženja kao i ideja okupljanja korisnika oko obrazovnih ciljeva utjecala je na daljnji razvoj ideje PLE-a [119]. Iste godine razvijena je prva verzija *Elgg* platforme koja omogućuje izgradnju vlastite društvene mreže u koju autor može uključiti različite Web 2.0 alate i servise koji mu mogu pomoći u učenju. Na JISC/CETIS konferenciji o alatima, standardima i sustavima za elektroničko učenje održanoj u Oxfordu 2004. godine, prvi put je spomenut pojam PLE. Ideja PLE-a koja je tom prilikom razmatrana bila je da to bude dodatni sloj unutar postojećih VLE-ova koji će povećati produktivnost korisnika olakšavajući pristup resursima smještenima van formalne institucije [74].

4.3.3 Osobni okoliš za učenje baziran na Web 2.0 servisima

Scott Wilson, istraživač na Sveučilištu u Boltonu 2005. godine objavio je konceptualni model virtualnog okruženja za učenje baziranog na Web 2.0 principima i alatima [87][88]. Zamislio je virtualno okruženje za učenje u središtu kojeg se nalazi učenik i koji ga koristi za prikupljanje i pohranu materijala, objavu vlastitih materijala i komunikaciju s drugim učenicima. To je središnje mjesto na kojem ima mogućnost uvida u sve svoje resurse bez obzira na njihov izvor. Za prikupljanje i pohranu materijala koristi dostupne Web 2.0 servise koji omogućuju jednostavan rad i suradnju u grupnom okruženju. Predloženi pristup izgradnje sustava za učenje povezivanjem postojećih servisa bio je zapažen i od tada se većina sustava za podršku stvaranju osobnog okoliša za učenje bazira na webu kao platformi i Web 2.0 servisima. Wilsonov model još je uvijek aktualan i čini polaznu točku u izradi brojnih sustava za stvaranje PLE-a.

Jedna od osnovnih funkcija PLE-a je da služi kao agregator resursa dostupnih na webu. Praktičari smatraju da za stvaranje osobnog okoliša nije potrebna nikakva posebna aplikacija pored onih koje već postoje na webu. Tako se PLE može stvoriti korištenjem bloga, wikija, okruženja društvenih mreža ili personaliziranih web stranica. Svatko može prema svojim željama izabrati onu vrstu aplikacije koja mu najviše odgovara. Učenici koji vole materijale poredane u kronološkom redoslijedu, izabrati će strukturu bloga dok će oni koji više vole razradu po ključnim pojmovima izabrati hijerarhijsku strukturu wiki sustava [8][27][52].

Osobni web portali prilagođeni korisniku poput *Pageflakesa* [URI27], *Netvibesa* [URI26] ili *iGooglea* [URI18] postaju naročito popularni pri učenju jer omogućuju jednostavnu izradu web stranice uklapanjem gotovih komponenti. Scott Wilson na Sveučilištu u Boltonu razvija mogućnost korištenja *widgeta* kao spone između klasičnih VLE-ova i PLE-ova [86], a na Sveučilištu u Manchesteru nakon nekoliko prototipova započela je izrada nove verzije sustava za izradu osobnog okoliša za učenje mPLE. Sustav je oblikovan kao monolitna web aplikacija koja koristi web servise, omogućuje korisnicima zadavanje osobnih ciljeva u grafičkom obliku, te pridruživanje materijala zadanim ciljevima [79][80][82].

4.3.4 Osobni okoliši za učenje u obliku hibridnih web aplikacija

Posljednji trend u izradi PLE-ova su hibridne web aplikacije koje koriste labavo povezane web servise. U sklopu projekta MUPPLE [URI23] razvija se sustav za stvaranje PLE-a korištenjem labavo povezanih web servisa. U trenutnoj fazi razvoja naglasak je stavljen na izradu programskog jezika za krajnje korisnike pomoću kojeg bi oni mogli samostalno birati, podešavati i povezivati dostupne servise [85]. Graham Attwell razvija PLE za potrebe zaposlenika ureda za zapošljavanje u Kentu u Velikoj Britaniji [4], a u Njemačkoj na Sveučilištu u Aachenu izrađena je prva verzija PLEF okruženje za stvaranje osobnog okoliša za učenje agregiranjem materijala dostupnih na webu u elektroničkom obliku [13].

4.4 Usporedba sustava za stvaranje osobnog okoliša za učenje

U ovom dijelu usporediti će se sustavi za stvaranje osobnog okoliša bazirani na korištenju postojećih Web 2.0 alata (blog, wiki i personalizirane web stranice) s mPLE i PLEF sustavima.

Kada se promatra struktura PLE-a ne treba zaboraviti da je to *osobno* okruženje prilagođeno autoru. Način organizacije sadržaja koji je opisan u ovom poglavlju odgovara autoru magistarskog rada, a nekome drugome će možda više odgovarati drugačija organizacija.

4.4.1 Web 2.0 aplikacije kao agregatori sadržaja

Web je platforma na kojoj se danas nalaze brojne aplikacije koje se mogu koristiti kao podrška stvaranju PLE-a. Aplikacije te vrste vrlo su jednostavne za korištenje, a većina ih je dostupna besplatno. One omogućuju pohranjivanje i povezivanje različitih materijala dostupnih na webu. Najčešće se povezivanje radi jednostavnim umetanjem URI-ja određenog sadržaja ili preuzimanjem objavljenog sadržaja u nekom od formata za objavu.

4.4.1.1 Blog i wiki sustavi

Blogovi i wikiji su web aplikacije za jednostavno uređivanje sadržaja. Po načinu korištenja i svojim mogućnostima te dvije vrste aplikacija su vrlo slične, a osnovna razlika je u načinu organizacije sadržaja. Blog se bazira na kronološkoj evidenciji podataka, a wiki na tematskoj. Današnji blog i wiki sustavi dostupni na webu poput *Bloggera* [URI6], *Wordpressa* [URI40] ili *Wikispacea* [URI39] mogu koristiti funkcionalnost brojnih *widgeta*. *Widgeti* se u tim

sustavima koriste npr. za kreiranje oblaka korisničkih oznaka ili prikaz kataloga slika pohranjenih na nekom web servisu.

Osnovna namjena tih sustava je unos tekstualnih sadržaja u HTML obliku, iako se u sadržaj mogu umetati i materijali drugih vrsta poput slika, video ili audio zapisa. Da bi se omogućio unos obogaćenog sadržaja na odgovarajuće mjesto u tekstu potrebno je umetnuti odgovarajući HTML kod i konkretan sadržaj će se prikazati u okruženju članka. Većina dostupnih blogova i wiki sustava podržava stvaranje folksonomije i omogućuje pretraživanje sadržaja po kreiranim korisničkim oznakama ili pomoću klasičnog pretraživača. Mnogi sustavi za kreiranje blogova su davatelji elektroničkog identiteta *OpenID* protokolom. Na takav način se rješava problem višestruke prijave pri korištenju različitih servisa. Prijavom na blog korisnik potvrđuje svoj identitet i nakon toga s tim istim identitetom može bez ponovne prijave koristiti druge web servise koji podržavaju *OpenID* protokol.

Osnovna prednost korištenja blog i wiki sustava kao agregatora je jednostavnost korištenja i dostupnost. Svatko doslovce u deset minuta može stvoriti svoj osobni okoliš za učenje i početi ga popunjavati sadržajem. Objavljeni sadržaji mogu se učiniti dostupnima drugim korisnicima weba u obliku RSS *feedova*, a brojni *widgeti* povećavaju njihovu osnovnu funkcionalnost.

Nedostatak korištenja je što su ti sustavi prvenstveno aplikacije za pohranu statičnih sadržaja. Sadržaji se označuju korisničkim oznakama dok je naprednije povezivanje sadržaja u takvom okruženju teško moguće.

Na slici 4.2 prikazan je primjer moguće organizacije sadržaja korištenjem bloga kao agregatora.

Slika 4.2 Blog kao agregator sadržaja

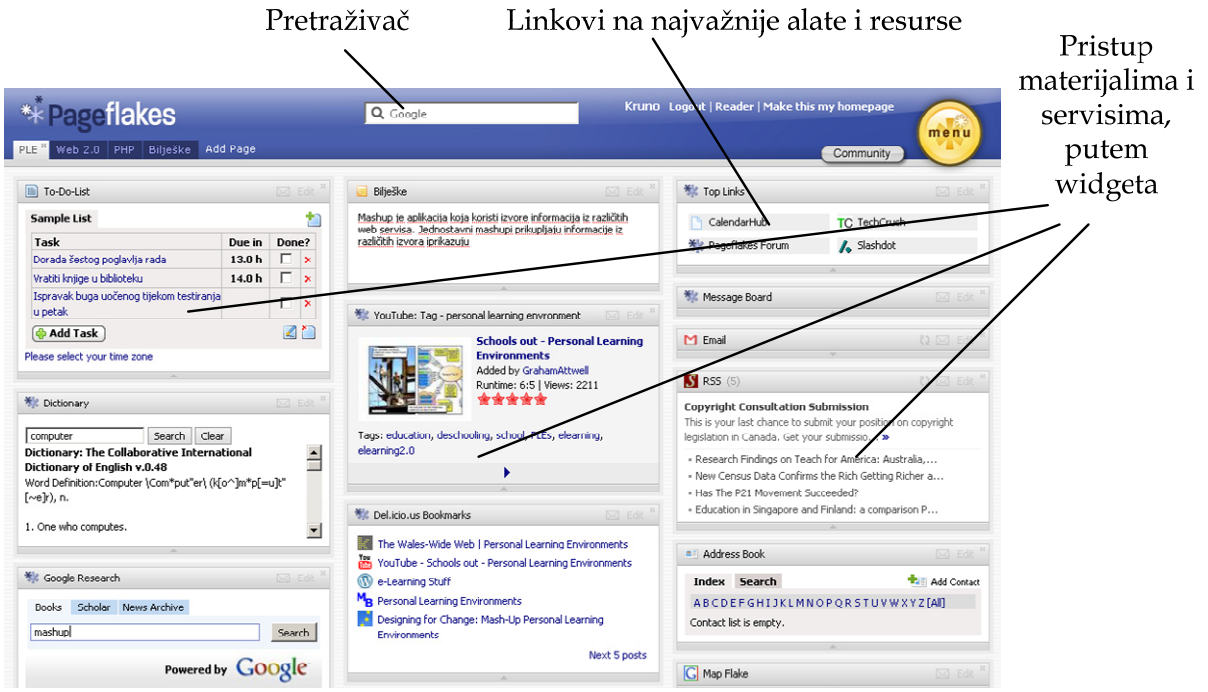
4.4.1.2 Osobni web portali bazirani na *Ajax* tehnologiji

Osobni web portali omogućavaju korisniku uključivanje različitih statičnih i dinamičnih sadržaja na web stranicu bez programiranja i potrebe za poznavanjem HTML jezika. Tehnologija na kojoj se baziraju je *Ajax*, a sadržaji koji se uključuju oblikovani su kao *widgeti*. Dio uključenih sadržaja može ostati privatan i dostupan samo korisniku, a dio se može učiniti javno dostupnim svima ili samo određenoj grupi korisnika. Po svojoj funkcionalnosti nema velike razlike između klasične web stranice i web stranice izrađene na ovakav način dok osnovna razlika proizlazi iz načina njezinog stvaranja. Da bi stvorio klasičnu web stranicu slične funkcionalnosti, korisnik mora poznavati HTML, CSS i JS tehnologije. Problem kod tog pristupa je što je proces izrade dugotrajan. Korištenjem osobnih portala baziranih na *widgetima*, osobna stranica se može složiti u nekoliko minuta.

Kreirana web stranica predstavlja hibridnu web aplikaciju koja sadrži različite materijale prikupljene iz raznih izvora. U kreiranu stranicu jednostavno je umetnuti RSS *feed* u bilo kojem formatu, linkove na web stranice, video, audio ili slike. Korisnik može pohranjivati

vlastite materijale u tekstualnom ili HTML formatu i razmjenjivati poruke s drugim korisnicima.

Na slici 4.3. prikazana je struktura osobnog portala realiziranog *Ajax* tehnologijom korištenjem *widgeta*.



Slika 4.3 Osobni web portal baziran na *Ajax* tehnologiji

Trenutni problem pri stvaranju takvih aplikacija je što još nije dogovoren standard za izradu *widgeta* koji bi omogućio da se kreirani *widgeti* koriste unutar različitih okruženja iako se standard razvija [117]. Korisnik na raspolaganju ima *widgete* kreirane za izabranu platformu koje ne može koristiti na drugim platformama (npr. *iGoogle widgete* ne može koristiti unutar *Pageflakes* okruženja). Ako korisnik samostalno stvara veću količinu materijala koje želi učiniti dostupnima drugima u obliku bilješki i sl., ili želi okruženje koje je orijentirano grupnom radu, bolji izbor je korištenje bloga ili wikija. Ako s druge strane želi jednostavan agregator sadržaja, a ne smeta mu to što je ograničen dostupnim *widgetima*, osobni web portali bazirani na *Ajax* tehnologiji su bolji izbor.

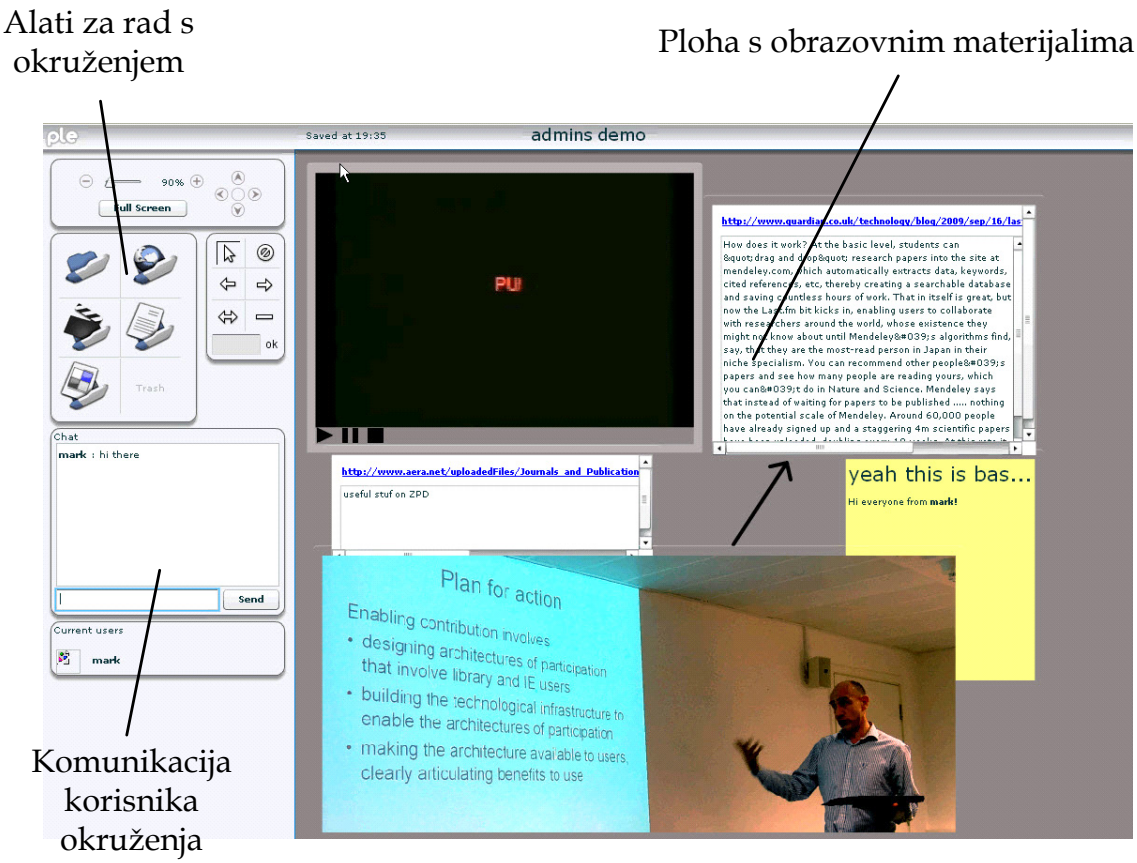
4.4.2 mPLE

mPLE je aplikacija za stvaranje PLE-a baziranog na principima društvenih mreža i servisima dostupnima na webu. Razvoj sustava započet je na Sveučilištu u Manchesteru, a danas ga razvija tvrtka *Hedtek*. Aplikacija podržava elemente društvenih mreža stvarajući radne prostore unutar kojih se odvija grupni rad učenika, stvaraju i dorađuju materijali te se provodi komunikacija. Učenici na početku trebaju odrediti osobne ciljeve i na osnovu njih izraditi okvirni plan učenja. Nakon toga samostalno ili u grupama ostvaruju ciljeve slijedeći zacrtani plan. Materijale stvorene tijekom tog procesa, pridružuju dijelovima plana, i na takav način

stvaraju opće znanje koje ostaje zapisano u sustavu i dostupno autoru i drugim učenicima [79][82].

Plan učenja je predstavljen grafom u kojem se navode ciljevi i podciljevi. Tijekom procesa učenja učenici mogu mijenjati i prilagođavati plan novostečenim spoznajama. Uz dijelove plana navođenjem URI-ja mogu se pridruživati materijali u elektroničkom obliku, a plan s uključenim resursima predstavlja repozitorij znanja učenika o određenoj temi. Učenik tom repozitoriju može pristupati i naknadno te ga koristiti, nadopunjavati ili mijenjati ovisno o novo spoznanim činjenicama. Sustav omogućuje sinkronu i asinkronu komunikaciju. Asinkrona komunikacija odvija se razmjenom tekstualnih poruka i putem foruma. U okruženje je uključena mogućnost stvaranje tekstualnih sadržaja u obliku bloga, a planira se realizacija sinkrone komunikacija pomoću *Skype* [URI33] servisa koji omogućuje komunikaciju korisnika razmjenom tekstualnih, audio i video poruka putem Interneta.

Na slici 4.4 prikazano je okruženje mPLE sustava s uključenim obrazovnim resursima.



Slika 4.4 Plan učenja s pridruženim resursima u mPLE sustavu

mPLE sustav je realiziran kao web aplikacija u kojoj je klijentski dio izrađen kao RIA kombinacijom *Ajax* i *Adobe Flash* tehnologija. Za izvođenje nekih aktivnosti i pohranu podataka sustav koristi funkcionalnosti dostupnih web servisa

Prednost sustava je što je namijenjen samostalnom učenju, a omogućuje grupni rad. Polazna osnova učeniku je zacrtani plan u grafičkom obliku. Takav prikaz informacija većini učenika pomaže u lakšem shvaćanju veza i odnosa između elemenata. Materijali se vežu uz čvorove

kreiranog grafa što omogućuje njihovo kasnije jednostavno pronalaženje. Nedostaci sustava proizlaze iz rane razvojne faze u kojoj se nalazi i činjenice što omogućuje korištenje relativno malog broja web servisa. Sustav razvija komercijalna tvrtka tako da će se njegove funkcionalnosti s vremenom sigurno povećati.

4.4.3 PLEF

PLEF sustav [URI29] razvijen je na Sveučilištu u Aachenu u sklopu istraživanja o primjeni Web 2.0 tehnologija u sustavima za računalom podržano učenje. Sustav je hibridna web aplikacija koja omogućuje agregaciju podataka iz više izvora. Svaki podatak izložen je u obliku *widgeta*, a korisnik vrši prilagodbu podataka koji će se prikazati (npr. unosi naslov, opis, korisničke oznake potrebne za pristup servisu).

Učenik se u sustav prijavljuje pomoću *OpenIDja* tako da može pristupati svim servisima koji podržavaju rad s *OpenID* identitetom bez potrebe za ponovnom provjerom identiteta. Svaki pojedini sadržaj može biti javno dostupan svim korisnicima sustava ili privatan i dostupan samo autoru. Učenik ima uvid u sve vlastite materijale, kao i sve materijale koji su javno dostupni, a kreirali su ih drugi učenici. Materijale drugih učenika može pregledavati i komentirati dok druge akcije po njima nisu dozvoljene. Sustav podržava asinkronu komunikaciju između korisnika slanjem poruka po principu elektroničke pošte. Vlastite materijale korisnik može označavati korisničkim oznakama u koje uvid imaju i drugi korisnici sustava, a po tim oznakama je omogućen pregled i pretraživanje sadržaja.

Izgled stvorenog osobnog okoliša za učenje prikazan je na slici 4.5.

Slika 4.5 Osobni okoliš za učenje kreiran pomoću PLEF aplikacije

Po svojoj funkcionalnosti sustav je vrlo sličan osobnim portalima baziranima na korištenju *widgeta*. Broj dostupnih *widgeta* za ovo okruženje je znatno manji, što je očekivano jer se radi o aplikaciji koja je u početnoj fazi razvoja. U odnosu na osobne portale, novost je prijava korištenjem *OpenID* korisničkog identiteta čime se olakšava korištenje funkcionalnosti dijela web servisa, označavanje sadržaja korisničkim oznakama i mogućnost komentiranja materijala.

Sustav na trenutnom stupnju razvoja ne donosi mnogo novosti u odnosu na web portale. Planirani model sustava [13][15] ima mnogo više funkcionalnosti koje u trenutnoj fazi razvoja aplikacije nisu realizirane. Između ostalog se planira da budući sustav omogući ne samo korištenje gotovih *widgeta*, već stvaranje novih funkcionalnosti kombiniranjem postojećih web servisa pomoću njihovih API-ja.

4.4.4 Usporedba opisanih sustava za stvaranje PLE-a

U prethodna tri dijela ukratko su opisana četiri različita načina na koji se može realizirati osobni okoliš za učenje. U praksi su testirana dva sustava za stvaranje bloga *Blogger* i *Wordpress* i dva sustava u kojima su realizirani osobni portali *iGoogle* i *Pageflakes* te PLEF sustav. mPLE aplikacije nije testirana u praksi jer u trenutku pisanja nije bila dostupna za testiranje, već je njezina funkcionalnost utvrđena na osnovu dostupne dokumentacije.

U tablici 4.1 uspoređene su osnovne osobine testiranih sustava koje su utvrđenje čitanjem dostupne dokumentacije i praktičnim testiranjem.

Tablica 4.1 Usporedba sustava za stvaranje PLE-a

	Blog	**Osobni portal**	**mPLE**	**PLEF**
Tehnologija klijenta	*Ajax*	*Ajax*	*Ajax +Flash*	*Ajax*
Provjera identiteta	*OpenID*, vlastita	Vlastita	Vlastita	*OpenID*
Višestruki identiteti	Ne	Ne	Ne	Ne
Stvaranje korisničkih sadržaja	Blog	*Widget*	Blog	*Widget*
Pridruživanje multimedijalnih materijala	HTML, *Widget*	*Widget*	Da	*Widget*
RSS	*Widget*	Da	*Widget*	*Widget*
Komentiranje	Da	Ne	Da	Da
Korisničke oznake	Da	Ne	Da	Da
Plan učenja	Ne	Ne	Da	Ne

	Blog	Osobni portal	mPLE	PLEF
Asinkrona komunikacija	*Widget*	*Widget*	Da	Da
Sinkrona komunikacija	*Widget*	*Widget*	Da	Ne
Uvoz sadržaja	Da	Da	Ne	Ne
Izvoz sadržaja	Da	Da	Ne	Ne
Status	Funkcionalno	Funkcionalno	U fazi izrade	U fazi izrade

Sve navedene aplikacije smještene su na webu, a dominantna tehnologija klijentskog dijela je *Ajax*. Razlog je taj što se na takav način omogućuje jednostavno indeksiranje i pristup sadržaju od strane web pretraživača. *OpenID* način utvrđivanja identiteta je raširen, a dio sustava djeluju i kao davatelji identiteta čime pojednostavljuju korištenje servisa. Niti jedan od sustava koji podržavaju *OpenID* protokol ne omogućuje korisniku stvaranje više pseudonima što je nedostatak jer se u praksi prilikom prijave može dogoditi situacija da davatelj identiteta nije u funkciji. U takvoj situaciji korisnik neće moći koristiti funkcionalnost sustava.

Tekstualni sadržaji koje stvaraju korisnici uglavnom se pohranjuju u vlastitom okruženju korištenjem funkcionalnosti bloga ili *widgeta*, a pri stvaranju HTML sadržaja koriste se vizualni uređivači teksta. Multimedijalni materijali (slike, video, zvuk i sl.) umeću se u sadržaj pomoću posebnih *widgeta* ili umetanjem JS koda u HTML sadržaj. mPLE omogućuje direktno povezivanje takvih materijala uz dijelove plana. Funkcionalnost dohvaćanja RSS sadržaja podržavaju svi sustavi.

Komentiranje sadržaja i označavanje korisničkim oznakama omogućavaju sustavi koji su orijentirani prema društvenim platformama. Osobni portali su više orijentirani prema jednom korisniku, pa zbog toga ne omogućavaju komentiranje materijala drugih korisnika niti označavanje korisničkim oznakama. Većina izabranih okruženja bazirana je na povezivanju materijala izloženih u obliku *widgeta* unutar okruženja i nemaju mogućnost izrade plana učenja. Ono što se u njima može napraviti po tom pitanju je tekstualni opis plana. Sustav mPLE je po tom pitanju drugačiji i njegovi korisnici imaju mogućnost izrade plana učenja u grafičkom obliku. Stvoreni plan pored toga služi i kao agregator materijala prikupljenih tijekom procesa učenja.

Svi sustavi omogućavaju asinkronu komunikaciju korisnika slanjem poruka, a većina (osim PLEF sustava) omogućuje i sinkronu komunikaciju korištenjem komunikacije čavrljanjem. mPLE sustav najavljuje asinkronu komunikaciju pomoću *Skype* servisa u kojoj će pored razmjene tekstualnih poruka biti podržana razmjena govora i slike. Uvoz i izvoz sadržaja je podržan u blogovima i osobnim portalima, dok u drugim sustavima nije. Razlog je što su drugi sustavi još u fazi razvoja pa se može očekivati da će tu funkcionalnost uključiti u budućim verzijama.

Sustavi opće namjene (blog i osobni portali) već postoje na webu i mogu se odmah koristiti za stvaranje osobnog okoliša za učenje. Ostali razmatrani sustavi uglavnom se razvijaju na znanstvenim institucijama u sklopu projekata i većinom su još uvijek u fazi razvoja ideje ili izrade prototipova. Najavljeno je da će mPLE sustav u beta verziji biti dostupan krajem 2009. godine [80], a kada se to dogodi to će biti prvi specijalizirani sustav za stvaranje PLE-a dostupan za korištenje na webu.

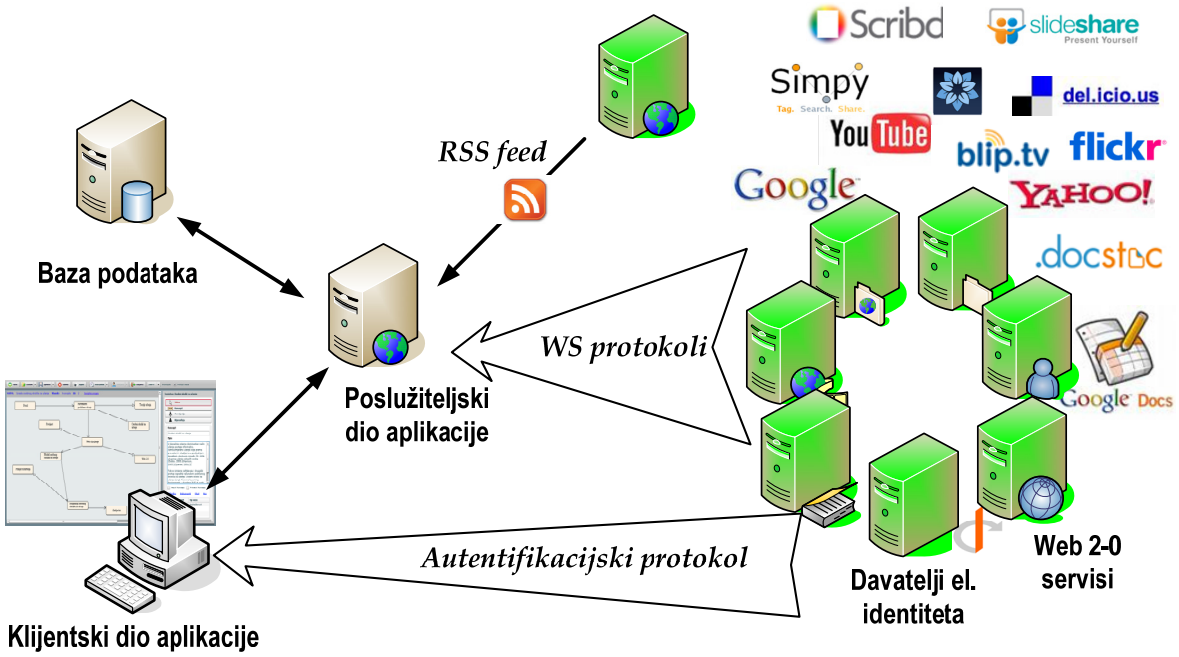
5 MODEL APLIKACIJE ZA PODRŠKU STVARANJU OSOBNOG OKOLIŠA ZA UČENJE

U ovom poglavlju opisan je model aplikacije za podršku stvaranju PLE-a. Model je osmišljen na osnovu teorijskih postavki opisanih u prethodnim poglavljima te osobnog iskustva pri samostalnom učenju korištenjem materijala i aplikacija dostupnih na webu. Funkcionalnost sustava bazirana je na idejama postojećih aplikacija i modela za stvaranje PLE-a [87][88][79][82][12]. U prva dva dijela ovog poglavlja opisani su opći zahtjevi koje sustav treba zadovoljavati. U trećem i četvrtom dijelu su razrađeni detaljni funkcionalni i nefunkcionalni zahtjevi, a u posljednjem je opisan logički model podataka.

5.1 Opći zahtjevi

Sustav je realiziran kao web aplikacija koja se sastoji od klijentskog i poslužiteljskog dijela. Poslužiteljski dio aplikacije koristi funkcionalnost servisa dostupnih na webu izloženu putem dostupnih API-ja. Korisnici aplikaciji pristupaju putem programa koji se izvodi u okruženju web preglednika. Provjera identiteta korisnika provoditi se pomoću distribuiranih protokola, a podržano je korištenje aplikacije na klijentskim računalima različitih platformi i operacijskih sustava.

Na slici 5.1 prikazan je model sustava u skladu s navedenim općim zahtjevima.



Slika 5.1 Model sustava

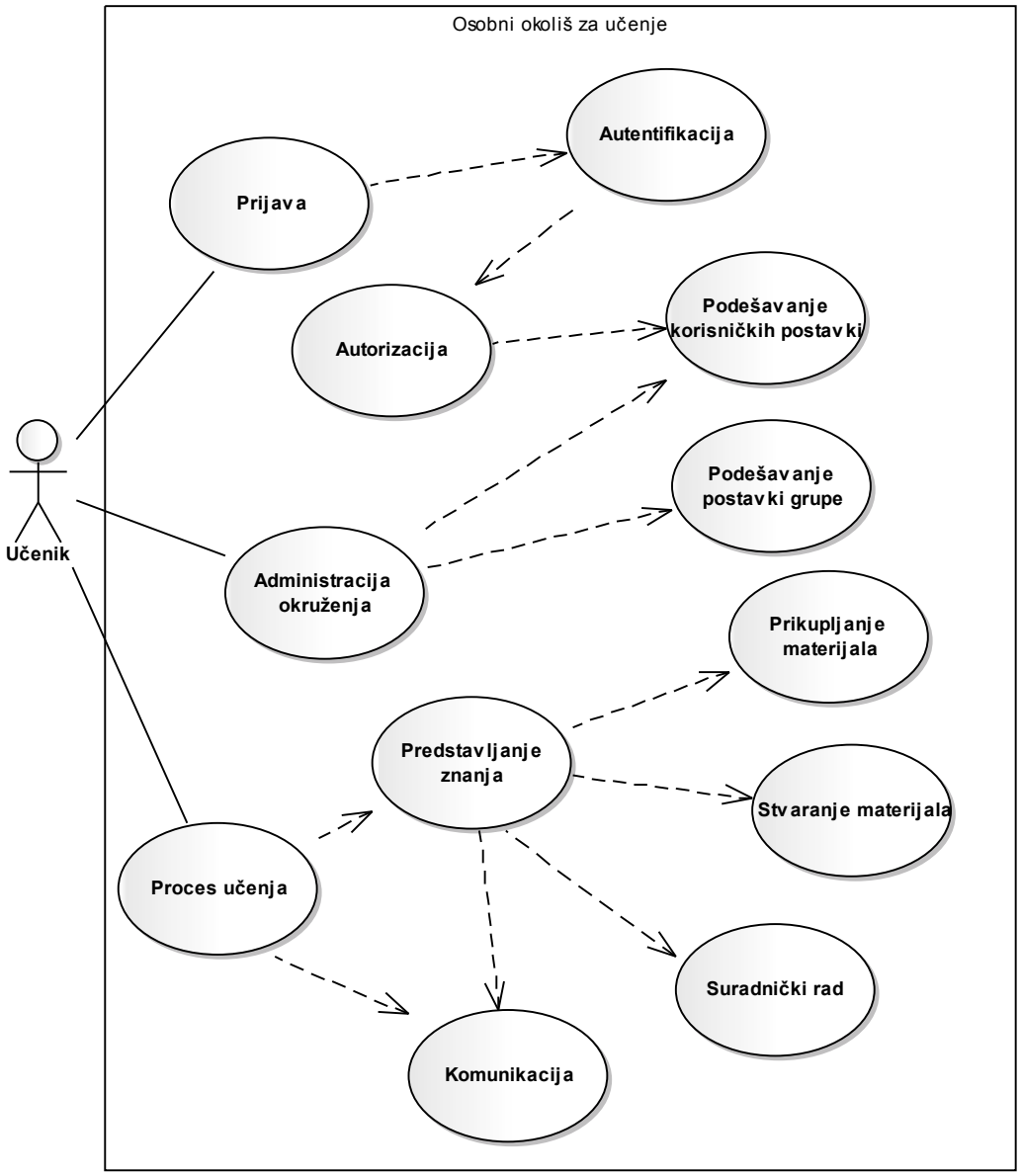
5.2 Opis potrebnih funkcionalnosti

Potencijalni učenik dolazi na početnu web stranicu sustava na kojoj se vrši provjera identiteta i prijava za rad sa sustavom. Ako je prijava bila uspješna, učenik dobiva pravo daljnjeg korištenja sustava. Nakon prijave učenik može obavljati administrativne aktivnosti koje nisu

vezane uz sam proces učenja ali su važne da bi se taj proces mogao odvijati na zadovoljavajući način. Te aktivnosti obuhvaćaju podešavanje okoliša vlastitim potrebama te evidenciju korisničkih podataka i grupnih postavki.

Osnovna namjena sustava je podrška procesu samostalnog učenja baziranom na postepenom stvaranju novoga znanja i predstavljanja stečenog znanja na prikladan način. Učenik tijekom procesa stvara znanje vršeći odgovarajuće aktivnosti i surađujući s drugim učenicima. Aktivnosti uključuju izradu vlastitih materijala, prikupljanja materijala dostupnih unutar okruženja ili na webu te komunikaciju s drugim učenicima. Navedeni procesi podržani su korištenjem web servisa.

Osnovna funkcionalnost koju sustav treba zadovoljavati opisana je UML dijagramom slučajeva korištenja prikazanom na slici 5.2.



Slika 5.2 Dijagram slučajeva korištenja

5.3 Specifikacija zahtjeva

U ovom poglavlju opisana je detaljna specifikacija zahtjeva za opisani model. Specifikacija je izrađena korištenjem FURPS+ metodologije i MoSCoW metode za označavanje prioriteta funkcionalnih zahtjeva.

5.3.1 Korištena metodologija

FURPS+ metodologija često se koristi za specifikaciju zahtjeva informacijskih sustava tijekom faze modeliranja i izrade. Naziv je akronim engleskih naziva za skupine zahtjeva koje obuhvaćaju funkcionalnost (engl *Functional*), iskoristivost (engl. *Usability*), pouzdanost (engl. *Reliability*) i učinkovitost (engl. *Performance*) sustava te mogućnost pružanja podrške (engl. *Supportability*). Pomoćni (engl. *ancillary*) „+“ u nazivu metodologije odnosi se na dodatna ograničenja na dizajn, izvedbu, izgled sučelja i fizičke zahtjeve sustava [45]. MoSCoW metoda se koristi za određivanje važnosti koja se pridaje izradi svakog pojedinog funkcionalnog zahtjeva pri isporukama programskog rješenja i važnost elemenata dijeli u četiri skupine [3]:

- M – funkcionalnost koja se obavezno mora realizirati (engl. *Must have*) da bi se isporuka smatrala uspješnom;
- S – funkcionalnost koju bi trebalo realizirati ako je ikako moguće (engl. *Should have*);
- C – funkcionalnost koja se može realizirati ako se time ne utječe na nedostatak neke druge (engl. *Could have*);
- W – funkcionalnost koja se neće realizirati u ovoj isporuci, ali je poželjna u slijedećima (engl. *Won't have this time*).

Pri određivanju prioriteta zahtjeva, zbog ograničenosti vremena vodilo se računa o tome da sustav ima osnovnu funkcionalnost koja se u slijedećim isporukama može inkrementalno nadograđivati do željene razine.

5.3.2 Funkcionalni zahtjevi

Funkcionalni zahtjevi podijeljeni po skupinama opisani su u nastavku ovog poglavlja. Uz svaki zahtjev, osim opisa i MoSCoW oznake naveden je i da li je realiziran u prvoj verziji aplikacije izrađene prema ovom modelu.

5.3.2.1 Prijava korisnika za rad na sustavu

Prijava korisnika za rad na sustavu je važna kako bi se osiguralo korištenje PLE-a samo autoriziranim osobama. Proces prijave se provodi u dva koraka. Prvi korak je provjera identiteta korisnika po distribuiranom modelu uz podršku nekog od vanjskih davatelja elektroničkog identiteta. Drugi korak slijedi nakon uspješne potvrde identiteta i obuhvaća

provjeru ovlasti za rad sa sustavom. Ako je korisnik prethodno evidentiran, omogućiti će mu se rad, a ako nije, prvo treba stvoriti svoj korisnički identitet unutar sustava.

U tablici 5.1 navedeni su detaljni funkcionalni zahtjevi vezani uz ovaj segment.

Tablica 5.1 Funkcionalni zahtjevi vezani uz segment prijave korisnika

Rb.	Opis	MoSCoW oznaka	Status realizacije
1.	Provjera identiteta osobe koja pristupa sustavu vrši se na decentralizirani način.	M	Realizirano
2.	Nakon uspješne provjere identiteta osobe sustav treba provjeriti da li je ta osoba evidentirana kao korisnik sustava. Ako je osoba korisnik sustava, potrebno je automatski dohvatiti korisničke postavke okruženja s poslužitelja, a ako osoba nije evidentirana, potrebno je omogućiti stvaranje novog korisnika.	M	Realizirano
3.	Nakon stvaranja novog korisnika potrebno je omogućiti automatsku prijavu novog korisnika u sustav bez ponovne provjere njegovog identiteta.	S	Realizirano
4.	Nakon dohvaćanja korisničkih postavki s poslužitelja, dohvaćene podatke je potrebno pohraniti u lokalnom okruženju klijenta. Pohranjene korisničke postavke ostaju zabilježene na klijentu sve dok se korisnik ne odjavi iz sustava.	S	Realizirano
5.	Ako u lokalnom okruženju klijenta postoje pohranjene korisničke postavke, pri dolasku na početnu stranicu okruženja potrebno je izvršiti automatsku prijavu u sustav onog korisnika čije postavke su zapisane.	S	Realizirano
6.	Prilikom odjave korisnika iz sustava, potrebno ga je automatski odjaviti iz okruženja davatelja elektroničkog identiteta.	W	Nije realizirano

5.3.2.2 Administracija okruženja

Administracija okruženja je segment koji obuhvaća sve aktivnosti koje nisu izravno vezane uz proces učenja ali su nužne da bi se taj proces mogao zadovoljavajuće odvijati. Te aktivnosti obuhvaćaju evidenciju korisničkih podataka i grupnih postavki, podešavanje web servisa koje sustav koristi te podešavanje izgleda okruženja sustava.

U tablici 5.2 prikazani su detaljni funkcionalni zahtjevi vezani uz ovaj segment aplikacije.

Tablica 5.2 Funkcionalni zahtjevi vezani uz segment administracije okruženja

Rb.	Opis	MoSCoW oznaka	Status realizacije
1.	Potrebno je omogućiti evidenciju osnovnih korisničkih podataka osobe: • korisničko ime; • ime i prezime; • kontakt podatke; • sliku; • bilješku o osobi; • korisnički identitet za provjeru.	M	Realizirano
2.	Nakon evidencije korisničkih podataka potrebno je vlasniku dozvoliti brisanje čitavog profila te izmjenu svih podataka osim korisničkog imena.	M	Realizirano
3.	Jednom korisniku potrebno je omogućiti evidenciju više različitih identiteta koje može koristiti prilikom prijave.	M	Realizirano
4.	Prilikom brisanja korisnika iz evidencije potrebno je obrisati sve njegove podatke: mape, koncepte i vezane materijale.	M	Realizirano
5.	Potrebno je omogućiti uspostavljanje veze između korisnika slanjem pozivnica. Prihvaćanjem poziva, korisnici stupaju u vezu. Ako korisnik odbije poziv, veza između korisnika neće biti uspostavljena. Korisnik treba imati uvid u status svih poslanih i primljenih pozivnica.	W	Nije realizirano
6.	Korisnik treba imati uvid u javno dostupne korisničke podatke svih drugih korisnika sustava.	S	Realizirano
7.	Korisnik treba imati mogućnost promjene vizualnih postavki sučelja (npr. fontova, boje, stila i sl.).	C	Nije realizirano
8.	Korisnik treba imati mogućnost izbora i podešavanja postavki sustava u koje se izvoze podaci.	W	Nije realizirano
9.	Korisnik treba imati mogućnost izbora i podešavanja postavki web servisa čiju funkcionalnost sustav koristi: • servisi za pretraživanje i pristup materijalima u elektroničkom obliku (npr. web stranica, slika, videa,…); • servisi za pohranu bilješki u tekstualnom i HTML formatu; • servisi za pohranu dokumenata (u pdf, doc, xls i arhivskim formatima); • servisi za asinkronu komunikaciju razmjenom poruka; • servisi za sinkronu komunikaciju.	M	Djelomično realizirano

5.3.2.3 Podrška procesu učenja

Ovaj dio predstavlja osnovnu funkcionalnost sustava. Učenik treba imati mogućnost izrade osobnog plana učenja i njegov prikaz u grafičkom obliku. Plan ima ulogu agregatora sadržaja i svakom od njegovih dijelova mogu se pridruživati materijali i rezultati aktivnosti.

U tablici 5.3 prikazani su detaljni funkcionalni zahtjevi koji se odnose na ovaj dio aplikacije.

Tablica 5.3 Funkcionalni zahtjevi vezani uz podsustav za podršku procesu učenja

Rb.	Opis	MoSCoW oznaka	Status realizacije
1.	Korisnik treba imati mogućnost stvaranja javne ili privatne mape u grafičkom obliku koja služi za prikaz plana učenja i vezanje materijala tijekom učenja. Mapa može biti javno dostupna svim korisnicima sustava ili privatna i dostupna samo autoru. Unutar mape potrebno je omogućiti uključivanje koncepata i njihovo međusobno povezivanje. Vlasnik mape ima potpune ovlasti nad mapom i njezinim elementima uključujući promjenu svojstava mape i brisanje mape dok drugi korisnici mogu pristupati elementima javno dostupne mape te ih koristiti i mijenjati.	M	Realizirano
2.	Potrebno je svim korisnicima sustava omogućiti ocjenjivanje kvalitete javnih mapa. Na osnovu dodijeljenih ocjena potrebno je omogućiti rangiranje mapa.	W	Nije realizirano
3.	Potrebno je svim korisnicima sustava omogućiti komentiranje javnih mapa.	C	Nije realizirano
4.	Potrebno je omogućiti izvoz mape i sadržaja u nekom od standardnih formata.	C	Djelomično realizirano
5.	Potrebno je omogućiti uvoz mape i sadržaja u nekom od standardnih formata.	C	Djelomično realizirano
6.	Potrebno je omogućiti uvoz materijala korištenjem web servisa.	W	Nije realizirano
7.	Potrebno je omogućiti izvoz i arhiviranje materijala korištenjem web servisa.	W	Nije realizirano
8.	Potrebno je omogućiti ispis mape i podataka o konceptima i vezama na pisač.	C	Djelomično realizirano
9.	Prilikom stvaranja mape, korisnik treba imati mogućnost stvaranja semantički različitih koncepata i veza. Takvi koncepti i veze trebaju se međusobno vizualno razlikovati (npr. različit oblik, veličina ili boja).	S	Nije realizirano
10.	Potrebno je omogućiti evidenciju veza koje imaju jedan ulaz, a više izlaza ili obratno.	C	Realizirano

Rb.	Opis	MoSCoW oznaka	Status realizacije
11.	Osim koncepata, unutar mape je potrebno omogućiti uključivanje druge mape. Preko te veze treba omogućiti pristup sadržaju vezane mape.	S	Nije realizirano
12.	Potrebno je omogućiti pretraživanje elemenata i sadržaja pohranjenih unutar mape. Po obavljenom pretraživanju, potrebno je omogućiti izravan pristup konceptima u kojima se nalazi traženi pojam.	M	Djelomično realizirano
13.	Potrebno je omogućiti unos svojstva svakom konceptu: • naziva; • opisa; • oznake privatnosti. Koncept može biti javni ili privatni. Privatni koncept dostupan je isključivo autoru koji ga je stvorio.	M	Realizirano
14.	Nakon evidencije koncepta potrebno je dozvoliti uvid, izmjenu ili brisanje podataka koncepta. Autor treba imati potpune ovlasti nad kreiranim konceptom uključujući njegovo brisanje. Ostali korisnici imaju pravo koristiti i mijenjati elemente javnog koncepta, ali ga ne smiju obrisati.	M	Realizirano
15.	Potrebno je omogućiti pridruživanje postavki koncepta njihovim kopiranjem iz drugog koncepta.	C	Djelomično realizirano
16.	Potrebno je omogućiti unos svojstva svake veze: • naziva; • opisa. Ako veza veže barem jedan javni koncept, ona je javna. U suprotnom je privatna i dostupna isključivo autoru.	M	Realizirano
17.	Nakon evidencije potrebno je dozvoliti uvid, izmjenu ili brisanje podataka veze. Vlasnik treba imati potpune ovlasti nad kreiranom vezom uključujući njezino brisanje.	M	Realizirano
18.	Potrebno je omogućiti pridruživanje postavki veze njihovim kopiranjem iz neke druge veza koja spaja dva koncepta.	W	Nije realizirano

Rb.	Opis	MoSCoW oznaka	Status realizacije
19.	Svakom konceptu potrebno je omogućiti pridruživanje različitih materijala u elektroničkom obliku (npr. web stranica, slika, video materijala,…). Pronađenim materijalima potrebno je omogućiti pridruživanje postavki: • naslova; • opisa; • korisničkih oznaka; • sličice sadržaja (engl. *thumbnail*); • oznake privatnosti.	M	Realizirano
20.	Nakon pohrane, potrebno je omogućiti uvid, izmjenu ili brisanje materijala vezanih uz koncept. Vlasnik ima potpune ovlasti nad pridruženim materijalima uključujući njihovo brisanje. Privatni materijali dostupni su isključivo autoru koji ih je stvorio ili uključio. Ostali korisnici imaju pravo koristiti i mijenjati podatke javnih materijala, ali ih ne smiju brisati.	M	Realizirano
21.	Potrebno je omogućiti pronalaženje materijala dostupnih na webu korištenjem standardnih web pretraživača (npr. *Google*, *Yahoo*,…).	M	Realizirano
22.	Potrebno je omogućiti pronalaženje materijala dostupnih na webu korištenjem pretraživača specifičnih servisa (npr. *Flickr*, *YouTube*,…).	M	Realizirano
23.	Potrebno je omogućiti istovremeno pretraživanje s više različitih pretraživača.	S	Realizirano
24.	Potrebno je omogućiti pohranu podataka vezanih materijala korištenjem web servisa.	S	Nije realizirano
25.	Potrebno je omogućiti pretplatu na vanjske RSS sadržaje.	M	Realizirano
26.	Potrebno je omogućiti pristupanje arhivi starijih RSS sadržaja.	W	Nije realizirano
27.	Svakom konceptu potrebno je omogućiti pridruživanje bilješki u tekstualnom i HTML formatu.	M	Djelomično realizirano
28.	Potrebno je omogućiti pohranu bilješki korištenjem web servisa.	M	Realizirano
29.	Korisnicima sustava je potrebno je omogućiti ocjenjivanje javno dostupnih materijala.	W	Nije realizirano
30.	Korisnicima sustava je potrebno omogućiti komentiranje javno dostupnih materijala.	S	Nije realizirano

Rb.	Opis	MoSCoW oznaka	Status realizacije
31.	Korisnicima sustava potrebno je omogućiti asinkronu komunikaciju razmjenom tekstualnih poruka.	M	Realizirano
32.	Korisnicima sustava potrebno je omogućiti sinkronu komunikaciju čavrljanjem. Korisnik treba imati mogućnost: • pratiti razgovor na određenu temu bez uključivanja u razgovor (pasivni sudionik); • sudjelovati u razgovoru (aktivni sudionik). Da bi postao aktivni sudionik korisnik se treba prijaviti u komunikacijski prostor. Po završetku komunikacije treba se odjaviti.	M	Realizirano
33.	Potrebno je omogućiti pohranu tijeka sinkrone komunikacije korištenjem web servisa.	S	Realizirano

5.3.3 Nefunkcionalni zahtjevi

5.3.3.1 Iskoristivost

Klijentski dio aplikacije realiziran je kao RIA čime se osigurava identičan izgled korisničkog sučelja na svakoj platformi. Kako bi sadržaji unutar sustava bili dostupni web pretraživačima, poželjno je da se sustav realizira u nekoj od tehnologija koje omogućuju indeksiranje sadržaja na jednostavan način. Kako bi sustav bio pogodniji za korištenje, poželjno je omogućiti korisnicima podešavanje vizualnih postavki okruženja (npr. povećavanje i smanjivanje veličine fontova, promjena veličine i pozicije elemenata, promjena boje i sl.) [60]. Izrada plana učenja i prikupljenog znanja realizirati će se u obliku konceptualne mape u kojoj će korisnici pomoću jednostavnog sučelja moći uređivati elemente.

Provjera identiteta korisnika treba se vršiti na decentralizirani način korištenjem *OpenID* protokola. Potrebno je omogućiti prijavu osobe korištenjem svih aktualnih verzija tog protokola.

5.3.3.2 Pouzdanost

Podaci koji se pohranjuju u okruženju poslužiteljskog dijela aplikacije čuvaju se u relacijskoj bazi podataka, pa se sigurnošću podataka bavi implementirani RDBMS.

Pri izboru web servisa za pohranu podataka u distribuiranom okruženju potrebno je voditi računa o izboru servisa koji svojim osobinama mogu u odgovarajućoj mjeri garantirati kvalitetu pristupa i korištenja pohranjenih podataka. Izbor treba provesti na osnovu rezultata ispitivanja osobina dostupnih servisa.

5.3.3.3 Učinkovitost

Pri izboru web servisa za dohvat podataka treba voditi računa o brzini pristupa podacima. Brzina pristupa web servisima ne smije utjecati na normalan rad korisnika. U fazi testiranja potrebno je donijeti odluku o izboru web servisa koji će se koristiti u obradama.

Odluku je potrebno donijeti na osnovu promatranja osobina web servisa tijekom testiranja u praksi na osnovu slijedećih parametara:

- *Dostupnosti* - prednost je potrebno dati besplatnim web servisima koji ne naplaćuju korištenje funkcionalnosti;
- *Stabilnosti rada* – vrijeme koje su servisi van pogona zbog greške ili održavanja trebalo bi biti što je moguće manje;
- *Stabilnosti izvedbe* – servisi ne bi smjeli često mijenjati svoje API sučelje jer svaka izmjena tog sučelja zahtjeva izmjene u programskom kodu;
- *Brzine rada* – servisi bi trebali u razumnom roku opsluživati zahtjeve korisnika. Kod servisa koji služe za pohranu podataka, posebno je potrebno obratiti pažnju na funkcionalnost zapisivanja podataka, jer je ona vremenski kritična operacija;
- *Jednostavnosti korištenja* – pri odluci o izboru servisa potrebno je voditi računa o protokolima koje njihov API koristi i prednost dati jednostavnijima (npr. XML-RPC ili REST u odnosu na SOAP).

Granica trajanja obrade nakon koje bi korisnika trebalo obavijestiti o tijeku obrade je oko jedne sekunde [59]. Poželjno je za vrijeme duljih obrada korisnicima omogućiti normalno obavljanje drugih aktivnosti. To se može učinkovito napraviti u okruženju RIA korištenjem asinkronog načina obrade. Tijekom duljih obrada korisnik bi kada god je to moguće trebao dobiti informaciju o očekivanom trajanju obrade. Kod obrada koje se odvijaju u lokalnom okruženju web poslužitelja i baze podataka, primjenom određenih modela takva informacija bi se mogla dosta dobro procijeniti [55]. Npr. prilikom prijenosa datoteka s klijenta na poslužitelj moguće je dosta precizno pratiti tijek prijenosa, pa bi tom prilikom u aplikaciji trebalo koristiti indikator koji prikazuje očekivano vrijeme trajanja prijenosa. U distribuiranom okruženju weba gdje se komunikacija odvija pozivanjem servisa smještenih na različitim web poslužiteljima, takvu procjenu je teže napraviti i u takvim slučajevima dovoljno je prikazati indikator koji prikazuje stanje obrade, ali ne i očekivano vrijeme njezinog trajanja.

Prilikom izbora protokola koji će se koristiti za komunikaciju između klijentskog i poslužiteljskog dijela aplikacije potrebno je voditi računa o količini podataka koja će se prenositi i o karakteristikama korištenog okruženja. U ovom okruženju poželjno je koristiti protokol koji svojim karakteristikama bolje podržava prijenos veće količine podataka.

5.3.3.4 Mogućnosti proširenja

Aplikacija se instalira na web poslužitelj, pristupa joj se putem web preglednika, a za pohranu podataka koristi funkcionalnost relacijske baze podataka. U aplikaciji je poželjno omogućiti korištenje različitih baza podataka.

U svom radu aplikacija koristi web servise koje korisnik treba moći samostalno izabrati i podesiti njihova svojstva. Osim toga potrebno je omogućiti proširivanje funkcionalnosti aplikacije korištenjem novih web servisa koji bi se u aplikaciju uključivali na jednostavan način.

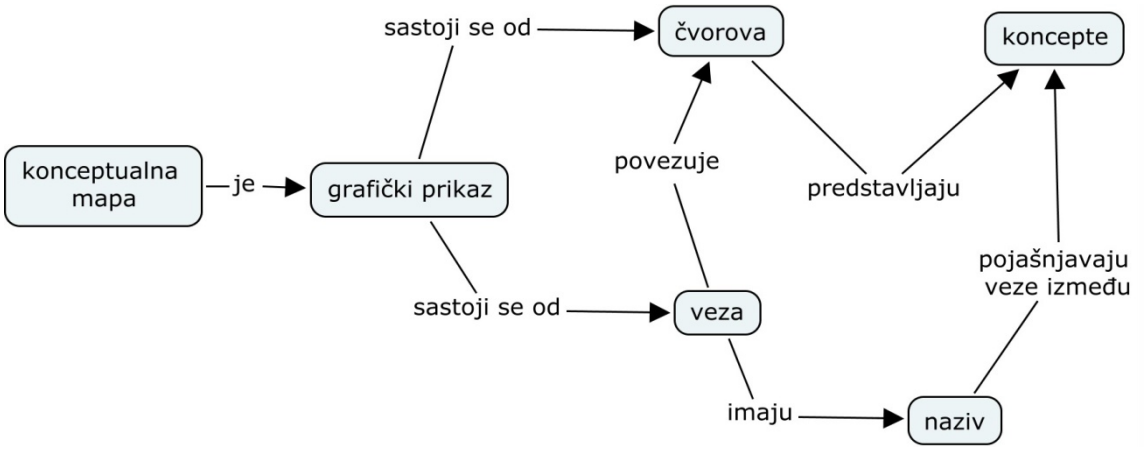
5.4 Analiza specifičnih zahtjeva

5.4.1 Izrada konceptualne mape

Učenici će za izradu plana učenja i predstavljanja strukture stečenog znanja koristiti konceptualne mape. Konceptualna mapa je grafički alat koji omogućuje strukturirani prikaz informacija, a obično se koristi za vizualno predočavanje informacija čime se omogućuje jednostavan uvid u odnose između koncepata. U osnovi, to je graf sastavljen od čvorova (koncepti) i veza među njima [61]. Prikaz informacija u takvom obliku može pomoći u lakšem shvaćanju veza i odnosa između vezanih elemenata učenicima kojima odgovara vizualan način učenja [47].

Koncept se obično predstavlja pravokutnikom ili elipsom unutar koje je naveden naziv koncepta. Veze među konceptima su predstavljene linijama na kojima se obično navode nazivi koji pojašnjavaju tip veze. Pošto se radi o grafičkim alatima koji služe za strukturiranje informacija, na jednoj mapi se ne bi trebalo nalaziti previše informacija jer time mape postaju nepregledne. Konceptualnu mapu najbolje je kreirati oko neke ključne fraze ili kao odgovor na pitanja.

Na slici 5.3 prikazan je primjer konceptualne mape.



Slika 5.3 Primjer konceptualne mape

Učenik na početku samostalnog učenja može skicirati mapu u kojoj se prikazuju grube činjenice koje su mu poznate na početku. Ona mu pomaže pri sagledavanju područja koja

mora detaljnije obraditi i njihovih međusobnih veza. Tijekom učenja učenik prikuplja nove činjenice, mijenja i nadopunjuje prethodno poznate, a paralelno s tim procesom upotpunjuje i mijenja kreiranu mapu.

Ako se svaki od koncepata poveže s određenim materijalima u kojima se nalaze informacije o njemu, uloga mape se nadograđuje i ona više nije samo grafički alat koji pomaže lakšem shvaćanju odnosa između koncepata, već postaje mapa znanja u kojoj se u kontekstu zadanog problema nalaze sve relevantne informacije [47].

5.4.2 Pohrana podataka

U svojoj osnovi PLE je repozitorij za pohranu podataka u elektroničkom obliku identificiranih svojim URI-jem. Osnovni zahtjev je da se svi sadržaji pohranjuju na zajedničkom mjestu kako bi bili lako dostupni s različitih lokacija. Kako bi se izabrao optimalan način, analizirana su tri različita načina pohrane podataka:

- korištenje web servisa namijenjenih pohrani knjižnih oznaka;
- korištenje web servisa namijenjenih pohrani strukturiranih podataka;
- pohrana u relacijsku bazu.

5.4.2.1 Pohrana podataka korištenjem web servisa za pohranu knjižnih oznaka

Pohrana korištenjem web servisa namijenjenih pohrani knjižnih oznaka (npr. *Delicious* [URI8] ili *Simpy* [URI32] servisi ili *mikroblogging* platforma *Tumblr* [URI35]) je jednostavan i prirodan način za pohranu podataka identificiranih URI-jem. Trenutno postoji veći broj takvih web aplikacija koje su besplatne za korištenje, a većina ih koristi jednostavan RESTful ili XML-RPC API koji omogućuje korištenje njihovih funkcionalnosti. Osnovni nedostatak njihovog korištenja proizlazi iz činjenice što su to servisi za pohranu knjižnih oznaka pri čijoj izradi je naglašena jednostavnost, pa im nedostaje dio funkcionalnosti potrebne u ovom okruženju. Npr. svaki od takvih servisa ima mogućnost pohrane osnovnih informacija o resursu: naslova, opisa, korisničkih oznaka i URI-ja. Jedan od zahtjeva u ovom modelu je da se uz svaki URI prikaže njegov *thumbnail*. Mjesto za pohranu tog podatka u okruženju takvih servisa ne postoji. Taj problem može se riješiti tako da se u tekst opisa doda poseban meta podatak koji predstavlja URI *thumbnaila*. Nakon što se određena knjižna oznaka dohvati, iz opisa bi se pročitao URI njezinog *thumbnaila*. Takav pristup se može koristiti pošto taj podatak nije ključan u pretraživanju.

Najjednostavniji način na koji se može pohraniti taj podatak je korištenjem standardnog HTML `img` elementa označenog posebnom vrijednošću atributa `id`. Prednost takvog načina je što bi se sličica mogla prikazati automatski u polju opisa.

```
<img id="thumb" src="uri-na-kojem-se-nalazi-slika" alt="thumbnail"/>
```

Nedostatak je što se u sadržaju opisa može slučajno naći slika s istom vrijednošću *id* atributa. Kako bi se ta opasnost umanjila, za oznaku `idja` bi trebalo izabrati neku kombinaciju slučajno izabranih slova i brojeva za koju postoji mala vjerojatnost da će se ponoviti.

Drugi način koji se može koristiti za pohranu URI-ja *thumbnaila* je jednostavna sintaksa slična onoj koju koriste wiki sustavi za prikaz slika.

```
[[slika: uri-na-kojem-se-nalazi-slika]]
```

Ovakvo rješenje je bolje za korištenje kod onih web servisa koji imaju problema s korištenjem HTML i XML oznaka u sadržaju, pa ih automatski uklanjaju.

API-ji koji se koriste za pristup web servisima ovoga tipa obično su vrlo jednostavni i imaju osnovne mogućnosti čitanja podataka po oznakama i zadanom uvjetu za pretraživanje te ažuriranja i brisanja podataka.
Prednost ovakvog načina pohrane je što se svim pohranjenim sadržajima može pristupiti i bez korištenja aplikacije putem sučelja web servisa. Preduvjet za korištenje je da korisnik prvo kreira korisnički račun na odgovarajućem web servisu. Nije potrebno inicijalno podešavanje web servisa, a ova vrsta servisa najčešće koristi jednostavnu provjeru identiteta korisnika pomoću korisničkog imena i lozinke.

Nedostatak ovakvog pristupa je u tome što ovaj tip web servisa nije predviđen za često ažuriranje sadržaja, pa oni korisnika obično ograničavaju u pristupu i zahtijevaju da korisnik podatke pohranjuje dulje vrijeme u *cacheu*. Pored toga vrlo mali broj takvih servisa podržava *OpenID* protokol.

5.4.2.2 Pohrana podataka korištenjem web servisa namijenjenih pohrani strukturiranih podataka

Funkcionalnost ove vrste web servisa (npr. *Google Docs* [URI15], *Zoho Creator* [URI44], *Amazon S3* [URI3]) vrlo je slična funkcionalnosti relacijske baze podataka. Podaci se pohranjuju u distribuiranom okruženju web servisa. Ako korisnik izabere ovakav način pohrane mora prvo kreirati korisnički račun na odgovarajućem web servisu. Da bi koristio funkcionalnost servisa mora nakon toga kreirati strukturu tablica koje će služiti za pohranu podataka. Pritom se definiraju sva polja u koja će se smještati podaci, njihova vrsta, veličina i mogućnost pretraživanja na način sličan stvaranju tablica u relacijskoj bazi podataka.

Broj aplikacija koje nude ovakav način pohrane podataka je dosta mali, a besplatni servisi obično su ograničeni količinom podataka koji se mogu unijeti i brojem korisnika koji mogu pristupati podacima (npr. besplatna verzija *Zoho Creatora* ograničena je na pohranu maksimalno tisuću slogova, a podacima mogu pristupati maksimalno dva različita korisnika). Ako je potreban unos većeg broja podataka ili korisnika mora se koristiti opcija koja se plaća.

Prednost ovih sustava je u tome što podržavaju osnovne CRUD funkcionalnosti na sličan način kao relacijske baze podataka. Zapisani podaci pohranjuju se na webu, a dostupni su kroz aplikaciju i kroz sučelje web servisa. Da bi korištenje ovih sustava za pohranu podataka bilo moguće, funkcionalnost prijave korisnika i stvaranja baze podataka obavezno bi trebalo realizirati kao dio administrativnog okoliša aplikacije za podršku stvaranju osobnog okoliša. Za korištenje u stvarnom radu i pohranu veće količine podataka potrebno je pretplatiti se na komercijalnu verziju zbog ograničenja koja postoje u besplatnoj verziji.

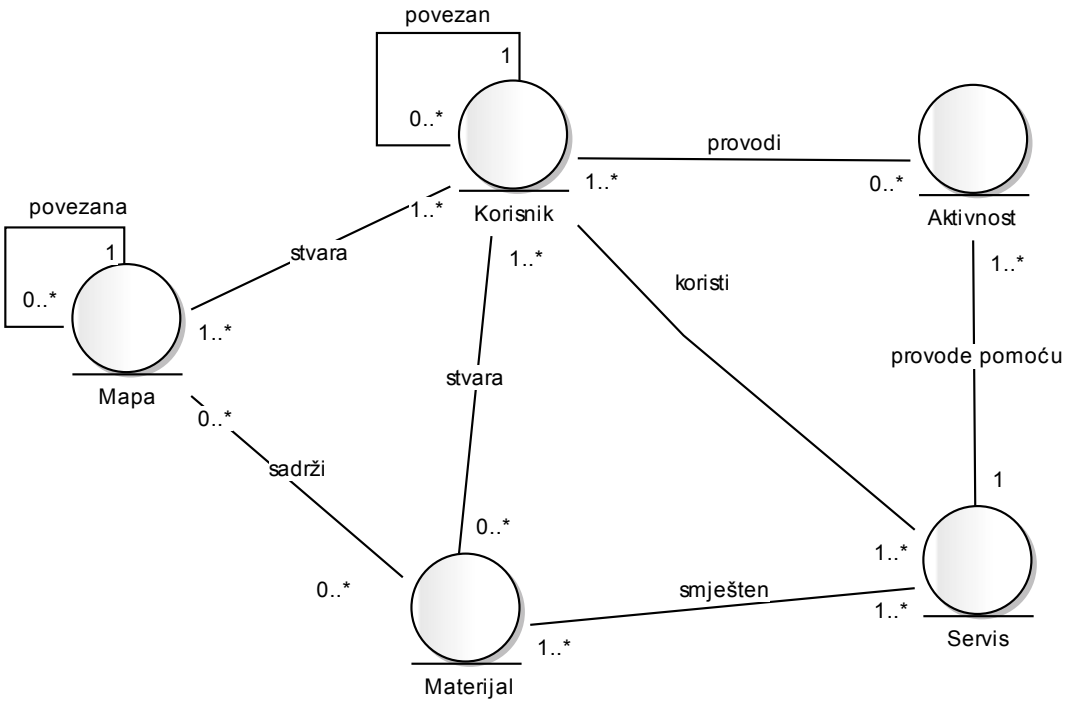
5.4.2.3 Pohrana podataka u relacijsku bazu

Sa stajališta izrade programa, pohrana podataka u relacijsku bazu je najjednostavniji način pohrane. Strukturu baze moguće je u potpunosti prilagoditi zahtjevima aplikacije i na takav način optimizirati korištenje. Relacijska baza ima sve potrebne osobine za sigurnu i učinkovitu pohranu podataka i brine se za održavanje integriteta podataka. Aplikacija koristi jednu bazu za sve korisnike, a oni nemaju potrebu za dodatnim podešavanjem. Osnovni nedostatak ovakvog načina pohrane je što korisnik podacima ne može pristupiti bez korištenja aplikacije.

Kako bi se korisnicima olakšalo korištenje podataka u slučaju prekida rada, u aplikaciju bi se mogla ugraditi funkcionalnost sigurnosne pohrane podataka koja će podatke koji se zapisuju u relacijsku bazu pohranjivati automatski i na neki drugi web servis (npr. kao knjižne oznake unutar nekog web servisa za pohranu knjižnih oznaka). Pohrana tih podataka izvela bi se u pozadini, asinkrono tako da ne opterećuje korisnika i ne usporava rad. Na takav način, korisnik bi imao mogućnost pristupa svim materijalima i van okruženja aplikacije.

5.5 Model podataka

Na slici 5.4 prikazan je logički model podataka izrađenog sustava.



Slika 5.4 Logički model podataka

Sustav obuhvaća pet skupina podataka. `Korisnički podaci` se odnose na sve osnovne podatke samog korisnika (npr. ime, prezime, adresa, nadimak, slika,…) i postavke okoliša za učenje (npr. korišteni servisi i korisnički podaci potrebni za pristup tim servisima, izgled korisničkog sučelja).

Korisnik može stvarati grupe vežući se s drugim korisnicima sustava. U okruženju grupe ili pojedinačno korisnik može provoditi različite `aktivnosti` poput komunikacije i zajedničkog rada.

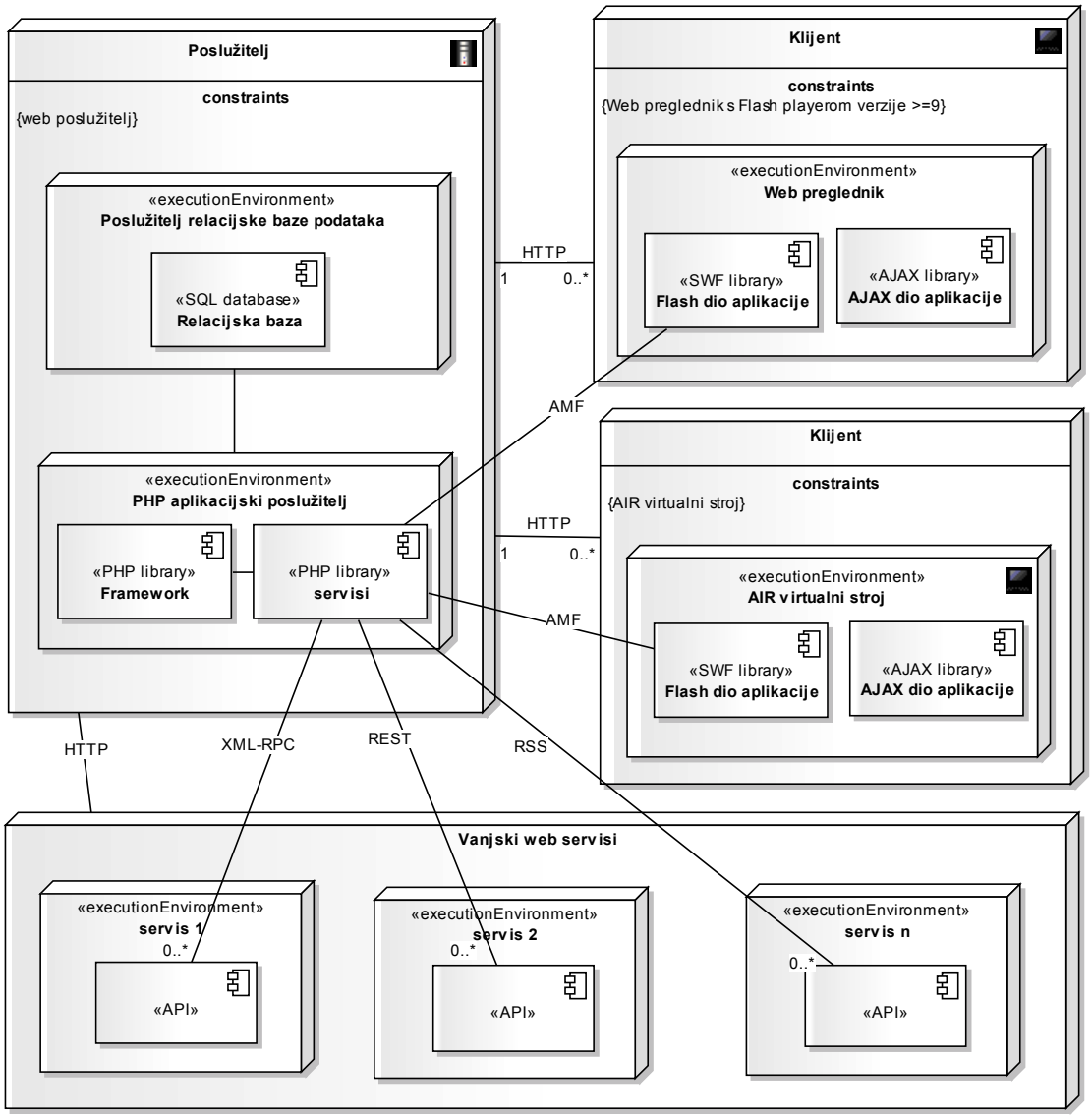
Svaki korisnik tijekom procesa učenja stvara `konceptualne mape`. Stvorena mapa može biti povezana s drugim mapama, a na jednoj mapi može zajednički raditi više korisnika. Dijelovima mape pridružuju se različiti `materijali` koje stvaraju korisnici koristeći funkcionalnosti distribuiranih `web servisa`.

6 REALIZACIJA APLIKACIJE ZA PODRŠKU STVARANJU OSOBNOG OKOLIŠA ZA UČENJE

Prema modelu opisanom u prethodnom poglavlju izrađena je web aplikacija *Samouk* koja podržava dio njegovih funkcionalnosti. U prvom dijelu ovog poglavlja opisana je arhitektura aplikacije koja je detaljnije razrađena u slijedeća tri dijela poglavlja. U posljednjem dijelu opisana su neka zanimljiva rješenja korištena tijekom razvoja.

6.1 Opis aplikacije

Na slici 6.1 prikazana je arhitektura aplikacije *Samouk*. Aplikacija se izvodi u web okruženju i sastoji od klijentskog i poslužiteljskog dijela. Poslužiteljski dio rađen je u PHP-u i zadužen je za većinu obrada i komunikaciju s web servisima. Za pohranu podataka koristi se relacijska baza *MySql*.



Slika 6.1 Arhitektura aplikacije *Samouk*

Klijentski dio poziva funkcionalnost poslužiteljskog dijela izloženu u obliku web servisa s kojima komunicira HTTP protokolom razmjenom RPC poruka u AMF (*Action Message Format*) [96] formatu. Klijentski dio izrađen je korištenjem *Adobe Flash* RIA tehnologije i izvodi se u okruženju web preglednika. Tehnologije su izabrane na osnovu svojih mogućnosti i jednostavnosti procesa razvoja. Pošto je poslužiteljski dio izrađen u obliku web servisa, njegovu funkcionalnost je moguće koristiti i od strane klijentskih aplikacija izrađenim drugim tehnologijama. Pri tome jedino ograničenje predstavlja korišteni AMF format.

Za pohranu podataka, pronalaženje i korištenje materijala u elektroničkom obliku koriste se različiti web servisi. Za komunikaciju s web servisima zadužen je poslužiteljski dio aplikacije na osnovu postavki koje je odredio korisnik. Komunikacija između poslužiteljske aplikacije i web servisa odvija se pomoću API-ja, a oblik komunikacije i korišteni protokoli ovise o konkretnoj realizaciji svakog pojedinog API-ja.

Prilikom izrade modela pretpostavilo se da će se podaci s web servisa dohvaćati na četiri različita načina: putem XML-RPC-a, RSS-a, REST-a i SOAP-a. U praksi se pokazalo da nema potrebe za izradom sučelja za komunikaciju pomoću SOAP protokola jer za korištenje nije izabran niti jedan servis koji ima samo SOAP API sučelje. Kod korištenih web servisa prevladavaju XML-RPC i REST oblici komunikacije, a u nekim slučajevima podaci se dohvaćaju u RSS formatu.

6.2 Klijentski dio aplikacije

6.2.1 Korištena tehnologija

Flash tehnologija za realizaciju klijentskog dijela izabrana je zbog jednostavnosti i brzine razvoja, te zbog toga što realizirana aplikacija ima jednaku funkcionalnost i izgled sučelja neovisno o korištenom operacijskom sustavu i web pregledniku. Pored toga, gotovo bez ikakve izmjene izvornog koda moguće je izraditi AIR aplikaciju koja se izvodi van okruženja web preglednika. Obje verzije u tom slučaju dijele istu početnu funkcionalnost, a u daljnjem razvoju AIR verzija se može proširiti dodatnom funkcionalnošću koju omogućuje platforma na kojoj se izvodi.

U razvoju je korišten programski jezik *ActionScript* verzije 3 (skraćeno AS3) i *Flex* programski okvir (engl. *framework*) koji omogućuje jednostavnu izradu *Flash* aplikacija korištenjem niza komponenti za izradu RIA. U manjem dijelu klijentska aplikacija koristi *Ajax* tehnologiju, i to za provođenje procesa provjere identiteta korisnika pomoću *OpenID* protokola.

Osnovna funkcionalnost koju klijentski dio aplikacije omogućuje je stvaranje mape znanja. Korisnik stvara mapu evidencijom koncepata i njihovim povezivanjem u smislenu cjelinu. Svakom konceptu pridružuje različite materijale u elektroničkom obliku koje dobavlja s weba preko poslužiteljskog dijela aplikacije. Osim materijala, korisnik može provoditi različite aktivnosti komunicirajući s drugim korisnicima na sinkroni i asinkroni način. Rezultati

komunikacije pohranjuju se u obliku bilješki vezanih uz odgovarajući koncept. Izrađena konceptualna mapa zajedno s pridruženim resursima predstavlja mapu znanja.

Flash aplikacija iz sigurnosnih razloga [34] ne podržava dohvat s različitih domena (engl. *cross-domain scripting*), osim ako poslužitelj s kojeg se podaci dohvaćaju to nije eksplicitno dozvolio i postavio datoteku u kojoj definira pravila takvog pristupa (engl. *cross-domain policy*) [99]. Velik broj Web 2.0 poslužitelja dozvoljava takav pristup, ali određen broj još uvijek ne, pa je zbog toga pri razvoju aplikacije korišten uobičajen pristup da podatke s web servisa dohvaća poslužitelj koji ih nakon toga prosljeđuje klijentu. Pošto se klijent i poslužitelj nalaze na istoj domeni, oni normalno komuniciraju.

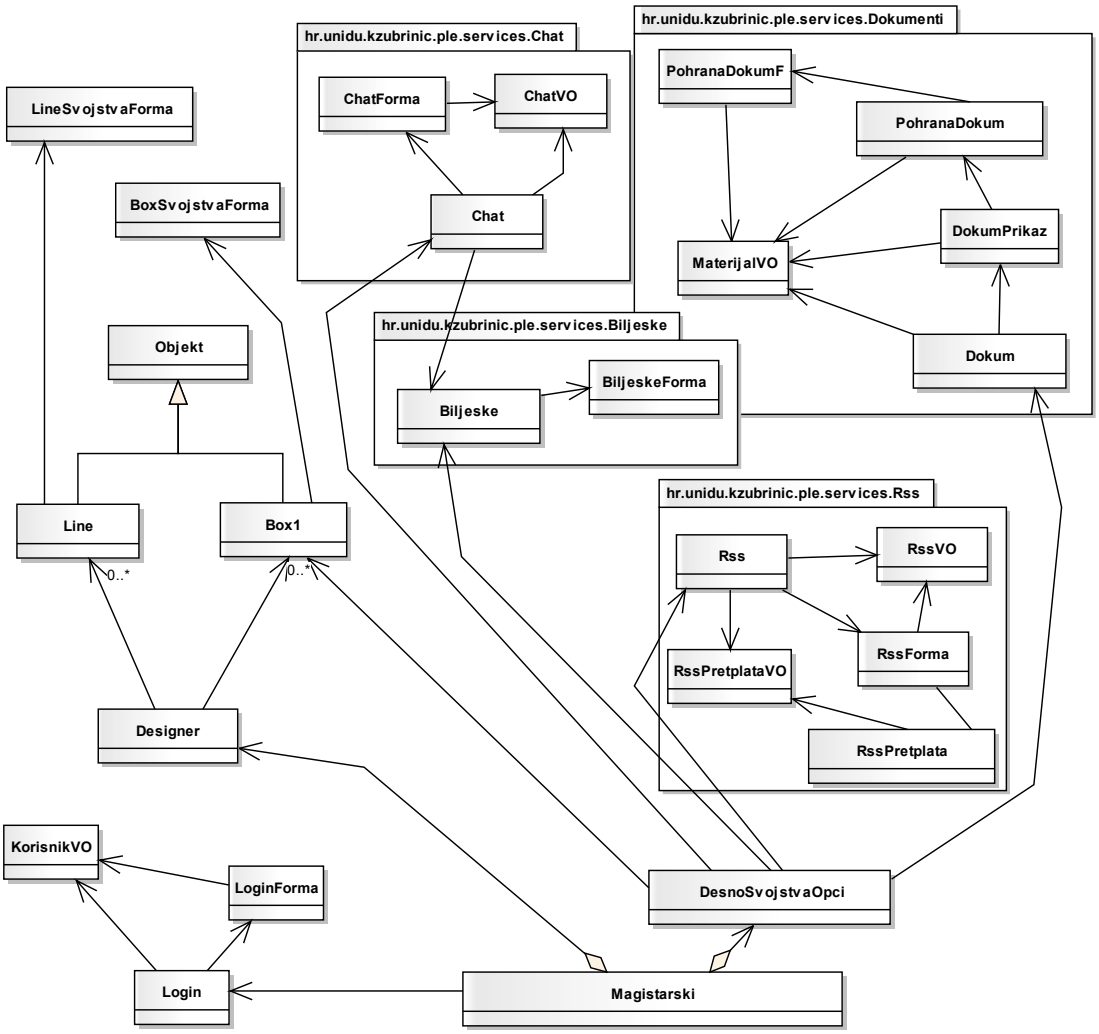
6.2.2 Struktura klijentskog dijela aplikacije

Aplikacija je oblikovana po principu model-pogled-upravljač (engl. *Model-View-Controller*, skraćeno MVC) arhitekture. MVC funkcionalnost aplikaciju dijeli na tri dijela [68]:

- *model* koji sadrži podatkovne elemente, odgovara na upite o vlastitom stanju ili mijenja stanje;
- *pogled* (engl. v*iew*) koji obuhvaća prezentacijski dio aplikacije odnosno grafičko sučelje na kojem korisnik vidi rezultate obrade;
- *upravljač* (engl. *controller*) koji obuhvaća dio koji je zadužen za izvođenje obrada i prosljeđivanje rezultata obrade modelu i prezentacijskom dijelu.

Polazna klasa aplikacije je `Magistarski` koja unutar sebe ima ukomponirana dva osnovna dijela. Prvi dio je predstavljen klasom `Designer` koja upravlja oblikovanjem konceptualne mape, a zadužena je za stvaranja koncepata i veza među njima. Drugi dio je klasa `DesnoSvojstvaOpci` zadužena za evidenciju svojstava izabranog koncepta, te za pridruživanje materijala i aktivnosti konceptu. Ta klasa vezana je funkcionalnošću s nizom klasa smještenih u četiri paketa: `Dokumenti`, `Rss`, `Biljeske` i `Chat`. Unutar svakog od tih paketa nalaze se klase zadužene za dobavljanje izabrane vrste materijala odnosno izvođenje odgovarajućih aktivnosti. U svakom od njih klase su povezane na isti način. Jedna klasa je zadužena za prikaz podataka putem korisničkog sučelja i ima funkciju pogleda (npr. klasa `DokumPrikaz`). Druga klasa je zadužena za dobavljanje podataka s poslužitelja putem web servisa (npr. klasa `Dokum`). Ta klasa inicira vezu s web poslužiteljem i poziva poslužiteljsku aplikaciju šaljući joj parametre. Ona se nalazi u ulozi upravljača. Nakon što poslužiteljska aplikacija vrati rezultate obrade, upravljač ih prosljeđuje klasi koja ima funkciju modela (npr. klasa `MaterijalVO`). Nakon što model primi rezultate obrade, klasa zadužena za prikaz podataka na korisničkom sučelju automatski osvježava sučelje pristiglim podacima.

Na slici 6.2 prikazan je UML dijagram najvažnijih klasa klijentskog dijela aplikacije. Prikazani su dijelovi koji služe za prijavu korisnika, stvaranje konceptualne mape, pridruživanje materijala i provođenje aktivnosti u sustavu. Zbog preglednosti, na dijagramu su navedene samo najvažnije klase i veze među njima.



Slika 6.2 UML dijagram najvažnijih klasa klijentskog dijela aplikacije *Samouk*

6.2.3 Prikaz podataka

Većina materijala vezanih uz koncepte na klijentskoj strani prikazuje se u tabličnom obliku gdje svaki redak predstavlja elemente jednog dohvaćenog resursa. Prednost takvog načina prikaza je u tome što se na jednom mjestu dobiva uvid u sve dohvaćene podatke, a osnovni nedostatak je što se u jednom retku dobiva uvid samo u osnovne podatke resursa. Da bi se vidjeli svi dostupni podaci potrebno je izabrati odgovarajući redak, nakon čega će se u poljima ispod tablice prikazati njegovi detaljni podaci. U svakom retku tablice prikazuje se naziv, opis i *thumbnail* koja ga vizualno predstavlja. Koncept korištenja *thumbnaila* za prikaz sadržaja odavno se koristi kod prikaza slika, a u posljednje vrijeme postao je popularan i pri prikazivanju sadržaja ostalih vrsti (npr. video sadržaja ili web stranice). Istraživanja ponašanja korisnika weba prilikom pretraživanja informacija pokazala su da je kombinacija teksta s

opisom sadržaja i *thumbanila* najbolja kombinacija koja im pomaže da pronađu traženu informaciju [24][90]. Za dohvaćanje *thumbnaila* koriste se Web 2.0 servisi koji koriste bazu *thumbnaila* web stranica. Ako korisnik zatraži sliku neke stranice koja postoji u bazi, servis će ju vratiti odmah, a ako stranica ne postoji u bazi servis će vratiti inicijalnu sličicu, koju će zamijeniti stvarnom nakon što posjeti tu web stranicu i *thumbnail* pohrani u bazu podataka. Ograničavajući faktor kod korištenja servisa te vrste je što većina besplatnih servisa dohvaća početnu stranicu na domeni, a ne konkretnu stranicu koja se nalazi dublje u hijerarhiji.

Kako bi se smanjila količina podataka koji se prenose između klijenta i poslužitelja te poslužitelja i web servisa, odjednom se dohvaća manji broj zapisa (klijent obično dohvaća deset po deset podataka). Korisnik pomoću klizača ili tipaka na tipkovnici prelazi s jednog retka u tablici na drugi i pregledava pristigle podatke. Kada se približi dnu tablici, na trećem retku od kraja, aplikacija započinje proces dohvaćanja slijedećeg skupa podataka. Razlog zbog čega se proces pokreće prije nego korisnik dođe do kraja tablice je taj što je, ako se korisnik dovoljno približi kraju tablice velika vjerojatnost da ga zanimaju novi podaci koji se trebaju dobaviti s poslužitelja. Za dohvat tih podataka potrebno je neko vrijeme i čekanje korisnika na pristup novom skupu podataka biti će kraće u tom slučaju nego ako se proces dobavljanja novih podataka započne tek kada korisnik dođe do kraja tablice.

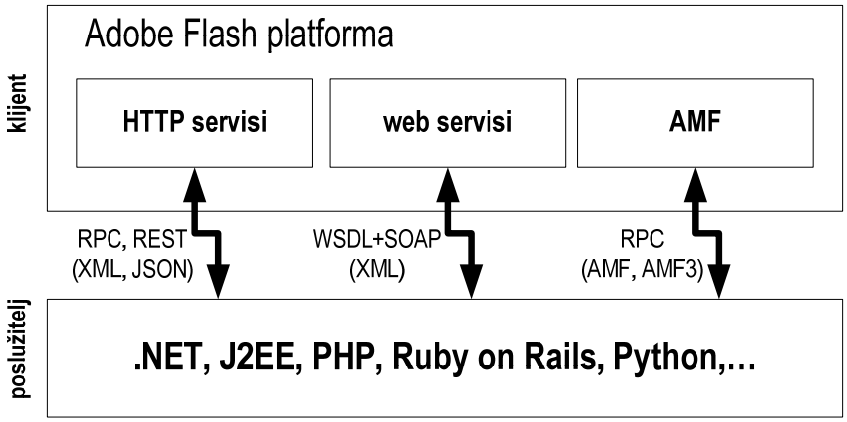
6.3 Komunikacija između klijenta i poslužitelja

6.3.1 Način prijenosa podataka između klijenta i poslužitelja

Komunikacija između klijenta i poslužitelja odvija se asinkrono što je uobičajeno kod korištenja RIA. Korisnik inicira akciju putem korisničkog sučelja, a klijent poziva odgovarajući servis na web poslužitelju predajući mu parametre. Nakon toga korisnik može nastaviti sa svojim ostalim aktivnostima.

Nakon što poslužitelj obradi zahtjev i vrati rezultate obrade, oni će se prikazati u klijentskoj aplikaciji. Aplikacija vizualno obavještava korisnika o tijeku trajanja asinkrone obrade pomoću animirane sličice koja prikazuje krug koji se vrti. Animacija se prikazuje u prozoru, obično ispod naslovne trake ili u alatnoj traci i vidljiva je sve dok obrada ne završi.

Između klijenta izrađenog *Flash* tehnologijom i web poslužitelja, podaci se mogu prenositi na tri različita načina prikazana na slici 6.3.



Slika 6.3 Načini komunikacije između *Adobe Flash* klijenta i poslužitelja

Prijenos u tekstualnom formatu korištenjem RPC ili REST pristupa je najjednostavniji za korištenje i na početku razvoja aplikacije razmatran je takav način pozivanja. Tijekom analize i probnog korištenja uočena su dva nedostatka. Prvi se tiče velike količina podataka koji se prenose, a drugi korištenja podataka i njihove serijalizacije. Prilikom prijenosa, često je potrebno prenositi veće količine podataka što dovodi do usporenja u radu zbog vremena koje je potrebno za prijenos podataka između klijenta i poslužitelja. Kada podaci dođu do klijenta, primljene podatke potrebno je serijalizirati odnosno pretvoriti u odgovarajući format koji se koristi unutar aplikacije. Pošto se klijentska aplikacija izvodi u okruženju web preglednika, pretvaranje veće količine podataka izaziva usporenje rada uslijed zauzeća veće količine memorijskih i procesorskih resursa na računalu klijenta.

Korištenje SOAP web servisa nije znatnije razmatrano zbog složenije realizacije i problema uočenih pri korištenju SOAP protokola i *Flash* okruženja.

Za prijenos je izabran RPC pristup s prijenosom podataka u binarnom AMF formatu [96]. Za razliku od tekstualnih protokola koji prenose podatke u generičkom obliku, pa je potrebno njihovo pretvaranje u odgovarajući format, AMF prenosi serijalizirane AS objekte, pa prilikom prijema nije potrebno njihovo dodatno pretvaranje. Osim toga zbog binarnog formata veličina prenesenih podataka manja, a brzina veća od protokola koji prenose tekstualne podatke. Format je izravno podržan od strane klijentske tehnologije, a pomoću dodatnih programskih biblioteka podržan je u nizu drugih programskih jezika koji se koriste za izradu poslužiteljskih web aplikacija (npr. PHP, *Java*, *Pyton*, .NET, *Ruby*).

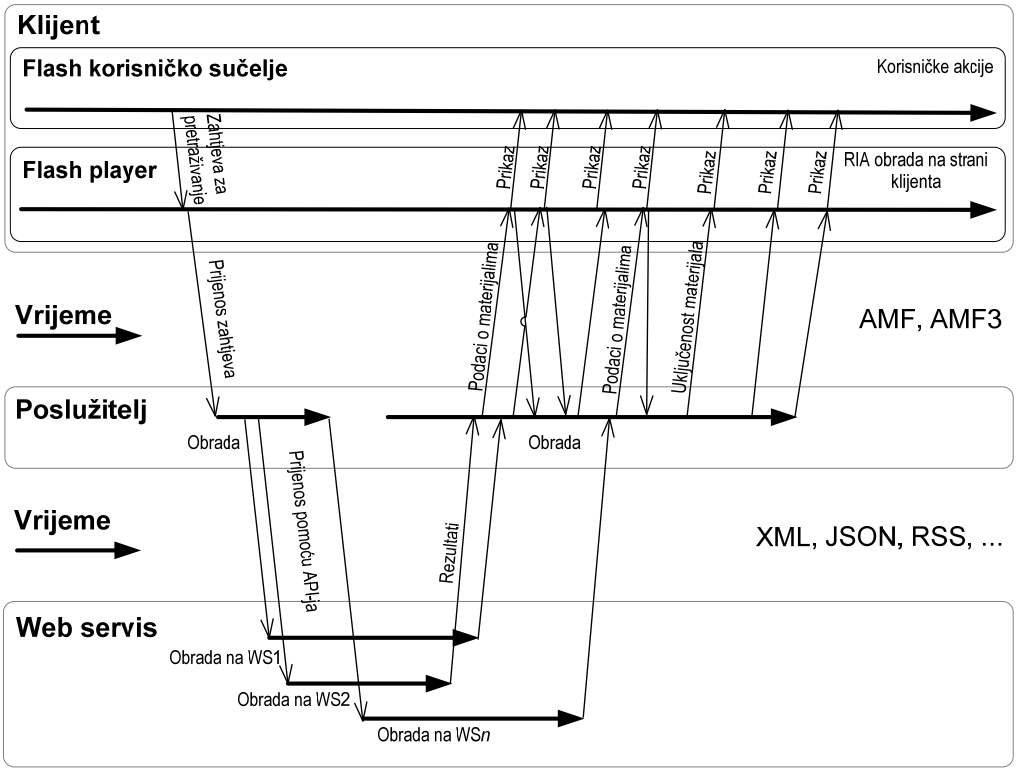
Npr. na klijentu je definirana klasa `KorisnikVO` koja opisuje strukturu podataka jednog korisnika. Kada se pomoću aplikacije unesu korisnički podaci novog korisnika, kreira se AS3 objekt toga tipa. Objekt se serijalizira i u AMF formatu šalje aplikaciji na poslužitelju. Na poslužitelju je definirana PHP klasa `KorisnikVO` čija struktura je identična strukturi klase `KorisnikVO` na klijentu. Kada metoda na poslužitelju primi objekt u AMF formatu vrši

njegovo automatsko pretvaranje u PHP objekt koji je preslika objekta koji postoji na strani klijenta.

6.3.2 Prijenos podataka o resursima

Najveći broj podataka koji se u sustavu prenose odnosi se na različite resurse koji se dobavljaju s weba i prikazuju u tabličnom obliku.

Na slici 6.4 prikazan je proces dohvaćanja podataka o resursima korištenjem web servisa.



Slika 6.4 Prikaz procesa prijenosa podataka o materijalima

Proces prijenosa inicira klijent tako da poslužitelju šalje zahtjev za određenim resursima. Aplikacija na poslužitelju korištenjem web servisa dobavlja tražene podatke te ih vraća klijentu. Nakon što se podaci prikažu u tabličnom obliku, za svaki od prikazanih materijala potrebno je dobaviti dodatnu informaciju da li je određeni materijal prethodno korišten ili nije. Ta informacija nije kritična za uspješno korištenje materijala, pa je za njezin dohvat izabran asinkroni način. Alternativa takvom načinu je čitanje te informacije odmah tijekom obrade rezultata primljenih od web servisa. Takav način bi produljilo vrijeme potrebno za prikaz konačnog rezultata obrade na zaslonu.

Korisnik putem korisničkog sučelja bira servise koje želi koristiti i unosi upit s ključnim riječima. Uneseni podaci se oblikuju kao AS3 objekt i šalju dijelu aplikacije na web poslužitelju. AMF komponenta na poslužitelju dohvaća poruku, kreira PHP objekt i pristupa ciljnim web servisima putem njihovog API-ja. Ako se koristi više web servisa, redoslijed pozivanja i dobivanja povratnih rezultata nije važan. Komunikacija s web servisima odvija se pomoću XML-RPC-a, REST-a i RSS-a ovisno o realizaciji API-ja konkretnog web servisa.

Web servis obrađuje primljeni zahtjev i vraća rezultat u formatu određenom API-jem. Program na poslužitelju čita primljene podatke i kreira PHP objekt. Ako je obrada uspješno provedena kreira objekt rezultata, a u suprotnom objekt greške. Kreirani objekt pomoću AMF komponente se vraća klijentu. Klijentski program dohvaća AMF poruku i učitava primljene podatke. Ako je primljen objekt greška, na zaslonu se prikazuje poruka o toj grešci, a u suprotnom se prikazuju rezultati u tabličnom obliku. Nakon prikaza rezultata korisnik može normalno koristiti primljene materijale, a program inicira niz asinkronih upita poslužitelju kako bi za svaki sadržaj dobio povratnu informaciju da li je uključen u bazu ili nije. Odmah po prijemu informacije, web poslužitelj prosljeđuje informacije klijentu koji osvježava prikaz onih materijala na koje se primljena informacija odnosi.

6.4 Poslužiteljski dio aplikacije

6.4.1 Korištena tehnologija

Poslužiteljski dio aplikacije izrađen je u PHP programskom jeziku, a za pohranu podataka korištena je *MySql* relacijska baza podataka. Aplikacija koristi vanjske Web 2.0 servise kao resurs za pohranu podataka te za pronalaženje i pristup materijalima u elektroničkom obliku.

Komunikacija s drugim web servisima odvija se korištenjem RPC, hibridnog REST i RESTful pristupa ovisno o realizaciji API-ja konkretnog servisa. U praksi Web 2.0 aplikacije najčešće koriste hibridni REST princip gdje se podaci dohvaćaju putem URI-ja, a kod ažuriranja i brisanja podataka koriste se pristup pozivanja funkcionalnosti metoda slanjem XML-RPC poruka.

Kao pomoć pri izradi funkcionalnosti korišten je *Zend Framework* [126] verzije 1.8.1 koji omogućuje jednostavniji i brži razvoj PHP aplikacija. Osnovna ideja programskog okvira je postojanje skupa modula, sučelja, apstraktnih i realiziranih klasa s gotovom funkcionalnošću često korištenih akcija (npr. upravljanje brzom privremenom memorijom (engl. *cache*), praćenje korisnika i stanja aplikacije, rad s relacijskim bazama podataka, web servisima i sl.). Prednosti korištenja programskog okvira u odnosu na klasičan razvoj očituju se u jednostavnijem i bržem razvoju.

Programski okvir skriva implementaciju konkretne relacijske baze od programera, tako da on ne mora voditi računa o razlikama u korištenju određene baze (načinu povezivanja, tipovima podataka, sintaksi SQL upita i sl.). Poslužiteljski dio aplikacije je konfiguriran za rad s *MySql* relacijskom bazom podataka, no pošto je u razvoju korišten programski okvir vrlo je jednostavno tu bazu podataka zamijeniti s bazom nekog drugog tipa bez velikih izmjena programskog koda aplikacije.

Kod korištenja web servisa programer ne mora brinuti o detaljima implementacije i svaki servis koristi na isti način: kreira objekt tipa web servis, a predaju parametara i dohvaćanje povratne vrijednosti vrši pozivom metoda na stvorenom objektu. Povratne vrijednosti se automatski pretvaraju u PHP objekt tako da programer ne mora voditi računa o konkretnoj realizaciji pojedinog servisa (da li su bazirani na XML-RPC, RESTful ili hibridnom pristupu)

te o formatu podataka (XML ili JSON). Takav pristup je vrlo koristan kod čitanja RSS podataka jer se podaci mogu objavljivati u *feedovima* različitih formata (RSS 0.9x, RSS 1.0, RSS 2.0, Atom). Svaki od formata ima različit oblik omotnice i strukturu sadržaja, a korištenjem funkcionalnosti programskog okvira podacima svakog *feeda* pristupa se na identičan način bez obzira na njegov format.

Klijentski i poslužiteljski dio povezani su razmjenom poruka u AMF formatu koji omogućuje izravnu razmjenu objekata, a *Zend Framework* pruža podršku korištenju AMF protokola unutar PHP programskog jezika.

Dohvat podataka pomoću web servisa je vremenski najzahtjevniji dio obrade. Podaci koji se dohvaćaju uglavnom se ne mijenjaju često, pa se nakon prvog dohvata spremaju u *cache* na web poslužitelju. Prilikom slijedećih čitanja dohvaćaju se podaci iz *cachea*, a ne pozivaju se web servisi. Mnogi web servisi, naročito oni koji su izrađeni po principima REST arhitekture potiču takav način rada i eksplicitno zahtijevaju da klijenti koji koriste njihove sadržaje, iste spremaju u svom lokalnom okruženju te ne dozvoljavaju česte upite od strane istog klijenta. Privremeno spremanje podataka obavlja se na disk web poslužitelja. U spremištu se podaci čuvaju određeno vrijeme, oko 2 sata. Kada klijent u okviru tog roka zatraži podatke, web poslužitelj će vratiti podatke iz *cachea*. Nakon što zadani vremenski rok prođe, upiti će ići prema web poslužitelju. Privremeno se pohranjuju sve vrste podataka koje se koriste u prijenosu. Fiksni podaci koji se unutar ove aplikacije ne mijenjaju (npr. HTML stranice, pdf dokumenti, video sadržaji ili slike) u *cacheu* se čuvaju fiksno zadano vrijeme. Podaci koji se mijenjaju unutar aplikacije (npr. bilješke) također se spremaju u *cache*, no kod svake izmjene sadržaja koja se napravi kroz program *cache* se briše.

Prilikom provjere identiteta korisnika *OpenID* protokolom posrednik u procesu je PHP skripta realizirana korištenjem *JanRain* PHP programske biblioteke [108]. Ta biblioteka je izabrana zato što je trenutno po svojoj funkcionalnosti najkompletnija i u potpunosti podržava sve trenutne verzije *OpenID* protokola. Uloga PHP posrednika u procesu provjere identiteta je oblikovanje poruke u skladu s *OpenID* protokolom pri prosljeđivanju na web stranicu davatelja identiteta te dekodiranje rezultata primljenog od strane *OpenID* davatelja identiteta po završetku procesa provjere

Ograničenje PHP jezika koje utječe na funkcionalnost aplikacije je što ne podržava tzv. *push* način rada [44] kod kojeg poslužitelj obavještava klijente o promjeni i inicira uspostavljanje veze. Zbog tog nedostatka u slučajevima kada je potrebna takva funkcionalnost (npr. kod sinkrone komunikacije putem poslužitelja) klijenti moraju češće slati upite poslužitelju što nije optimalno.

Nedostaci korištenja *Zend Frameworka* odnose se uglavnom na to što se razvija vrlo intenzivno pa su neke funkcionalnost još uvijek u ranoj fazi razvoja te se javljaju povremene greške, a dio funkcionalnosti nije dovoljno dobro dokumentiran. Drugi problem je što se povremeno između dvije verzije programskog okvira promjeni definicija sučelja za pristup funkcionalnostima, pa je tada potrebno mijenjati programe koji koriste tu funkcionalnost.

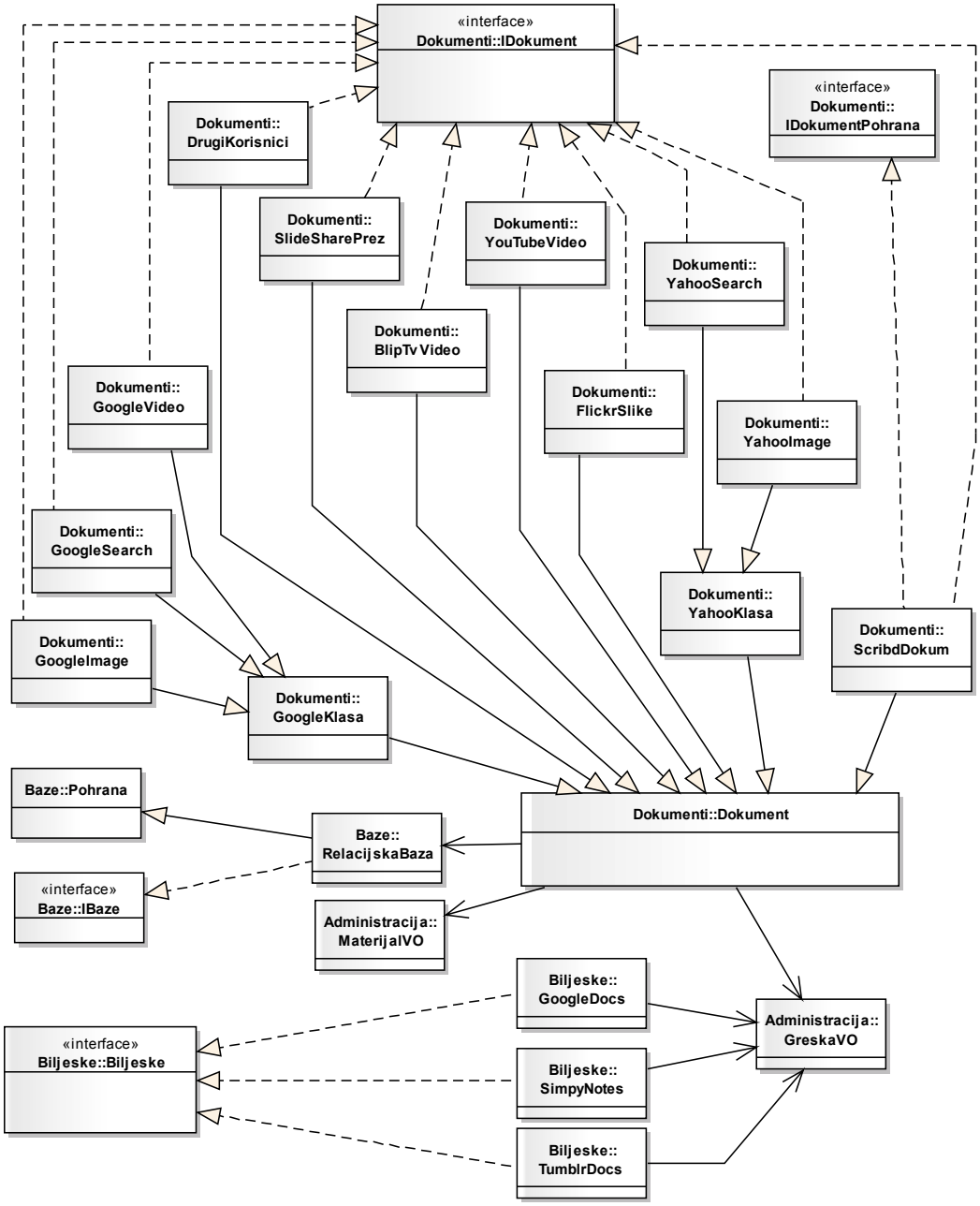
Objektna orijentacija *Zend Frameworka* omogućuje brži razvoj, ali izvođenje aplikacija je zbog toga nešto sporije.

6.4.2 Najvažniji segmenti poslužiteljskog dijela aplikacije

Aplikacija na poslužiteljskom dijelu je podijeljena na nekoliko dijelova prema funkcionalnosti, a u nastavku će se opisati najvažniji dijelovi aplikacije.

6.4.2.1 Evidencija obrazovnih materijala

UML dijagram klasa prikazan na slici 6.5 prikazuje međusoban odnos klasa i sučelja unutar paketa `Dokumenti`.



Slika 6.5 UML dijagram klasa paketa *Dokumenti*

U paketu `Dokumenti` smještene su sve klase zadužene za pristup obrazovnim materijalima na webu korištenjem web servisa. Materijalima se pristupa putem njihovog URI-ja pa je na takav način moguće pohraniti informaciju o svakom sadržaju koji se nalazi na webu (npr. web stranici, pdf dokumentu, slici ili video sadržaju). Za komunikaciju s web servisom koristi se API autora servisa koji određuje protokol i format podataka.

Program je organiziran na način da je za pozivanje svakog servisa potrebno kreirati po jednu klasu. Ime klase je proizvoljno, ali je poželjno da se zove slično kao servis na koji se odnosi. Svaka od tih klasa nasljeđuje klasu `Dokument` i implementira sučelje `IDokument`. U klasi `Dokument` nalazi se zajednička funkcionalnost svih klasa ovog paketa. Sučelje `IDokument` definira da svaka od klasa koja služi za vezu s određenim web servisom mora implementirati metodu `trazi` koja služi za slanje upita i dohvaćanje skupa rezultata. Metoda prima parametre potrebne za pristup web servisu, a vraća dohvaćene podatke u obliku polja objekata u kojem svaki objekt predstavlja jedan dohvaćeni mrežni resurs. Objekt sadrži id oznaku sadržaja, naziv, opis, URI sadržaja i URI *thumbnaila*. Ako se pri dohvaćanju podataka i obradi dogodi greška, stvoriti će se objekt greške.

Uključivanje funkcionalnosti novog web servisa u aplikaciju je jednostavno. Potrebno je kreirati novu PHP klasu koja nasljeđuje klasu `Dokument` i implementira sučelje `IDokument`. Kreiranu klasu je potrebno registrirati u datoteci `indeks.php` kako bi mogla koristiti funkcionalnost AMF formata, a u bazu podataka potrebno je zapisati podatke potrebne za korištenje novog servisa (npr. korisničko ime, lozinku, ključ i sl.). Vrsta podataka koje je u bazu potrebno zapisati ovisi o zahtjevima konkretnog API-ja.

6.4.2.2 Evidencija bilješki

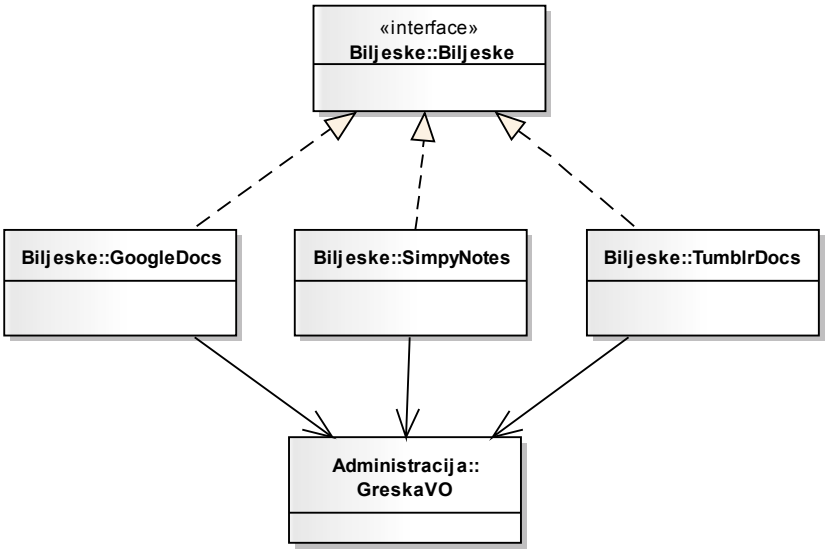
U paketu `Biljeske` nalaze se sve klase koje služe za evidenciju bilješki. Bilješke se pohranjuju pomoću web servisa specijaliziranih za pohranu podataka u tekstualnom obliku. Unutar klijentske aplikacije bilješke se mogu oblikovati u čistom tekstualnom ili HTML formatu. Za konkretnu realizaciju pohrane bilješki izabrana su tri web servisa. Da bi korisnik mogao koristiti zapis bilješki u bilo koje od tih okruženja, prvo mora stvoriti korisnički račun i podesiti ih svojim potrebama. Stvaranje korisničkih računa iz okruženja aplikacije u ovoj verziji aplikacije nije podržano, već korisnik mora otići na web stranicu određenog servisa i stvoriti svoje korisničke podatke. Nakon toga s tim podacima treba podesiti postavke aplikacije.

Google spreadsheets [URI17] je tablični kalkulator smješten na webu. Prilikom pohrane, podaci se smještaju u više tablica prema početnom slovu koncepta. Unutar pojedine tablice puni naziv koncepta služi kao ključ koji pokazuje na ćeliju u kojoj se nalazi tekst bilješke tog koncepta. Takav način realizacije je izabran zato što je unutar *Google* okruženja ograničen broj dokumenata koje korisnici mogu kreirati, pa pohrana svakog koncepta u poseban dokument nije moguća. Pošto ćelije tabličnog kalkulatora nisu ograničene količinom podataka koji se u njih mogu spremiti ovakav način pohrane zadovoljava potrebe.

Simpy je web servis za pohranu knjižnih oznaka koji ima dio za pohranu kraćih bilješki. Ograničenje koje nije opisano u dokumentaciji tog servisa već se na njega naišlo tijekom razvoja odnosi se na maksimalnu duljinu sadržaja koji se može pohraniti. Ono proizlazi iz tehničke realizacije *Simpy* API-ja koje za prijenos podataka pri pohrani koristi HTTP GET protokol, pa je količina podataka koji se mogu prenijeti ograničena karakteristikama tog protokola. U praksi, korištenjem *Simpy* web servisa mogu se zabilježiti kraće bilješke do maksimalno dvije tisuće znakova.

Tumblr je platforma za stvaranje mikro blogova koja omogućuje evidenciju različitih vrsti podataka (bilješki, linkova, slika i video sadržaja). U okviru aplikacije korištena je funkcionalnost za pohranu bilješki koje se pohranjuju po principu jedan koncept – jedna bilješka.

UML dijagram klasa na slici 6.6 prikazuje međusoban odnos klasa i sučelja unutar paketa `Biljeske`.



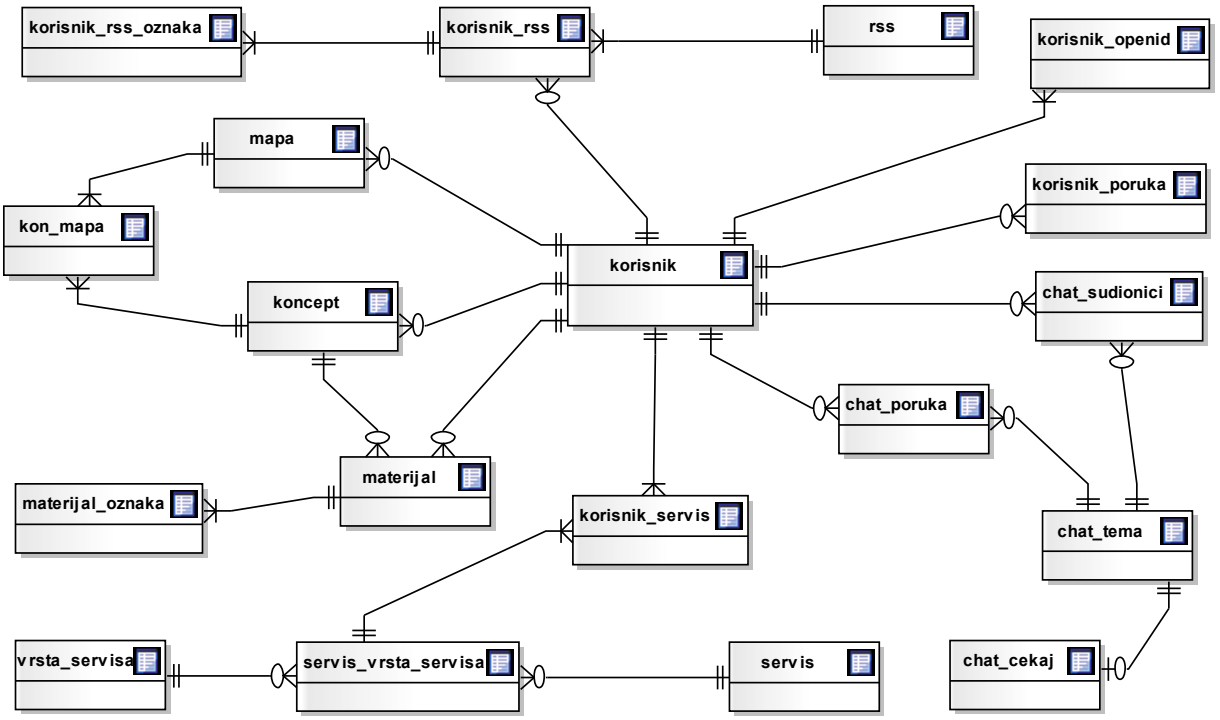
Slika 6.6 UML dijagram klasa paketa *Biljeske*

Organizacija programa unutar paketa za pohranu bilježaka slična je kao kod pohrane materijala. Svaka klasa implementira sučelje `IBiljeske` u kojem su navedene osnovne metode koje omogućavaju izvođenje funkcionalnosti dohvata, stvaranja, ažuriranja i brisanja podataka.

Novi servis za pohranu bilješki uključuje se tako da se kreira nova PHP klasa koja mora implementirati sučelje `IBiljeske`. Unutar klase navode se konkretne funkcionalnosti izabranog web servisa. Kako bi novo kreirana klasa mogla koristiti AMF format za prijenos podataka potrebno ju je registrirati u datoteci `indeks.php`, a u bazu podataka zapisati korisničke postavke potrebne za korištenje servisa.

6.4.3 Konceptualni model baze podataka

Na slici 6.7 prikazan je konceptualni model relacijska baze podataka koju koristi poslužiteljski dio aplikacije *Samouk*.



Slika 6.7 Konceptualni model relacijske baze podataka

Korisnički podaci se nalaze u tablicama `korisnik` i `korisnik_openid`. U ovim tablicama smješteni su osnovni podaci svih korisnika (ime, kontakt podaci, slika,…) i podaci potrebni za provjeru identiteta korisnika (*OpenID* oznake vezane uz korisnike). Svaki korisnik može imati više različitih *OpenID* oznaka, a jedna *OpenID* oznaka može se pojaviti u tablici samo jednom.

U tablicama `vrsta_servisa`, `servis`, `servis_vrsta_servisa` i `korisnik_servis` smješteni su administrativni podaci svih web servisa koje aplikacija koristi za pribavljanje i pohranu podataka. Podaci u tablicama `vrsta_servisa`, `servis` i `servis_vrsta_servisa` zajednički su za sve korisnike aplikacije i u njima se nalaze podaci o svim servisima s kojima aplikacija može komunicirati. U tablici `korisnik_servisi` nalaze se postavke onih servisa koje je određeni korisnik izabrao za korištenje.

Tablice `materijal` i `materijal_oznaka` služe za pohranjivanje podataka o svim resursima koje je korisnik prikupio. Resursi su vezani uz određenog korisnika i koncept, a javno dostupnim materijalima mogu osim autora pristupati i drugi korisnici aplikacije.

Podaci konceptualnih mapa smješteni su u tablicama `koncept`, `mapa` i `kon_mapa`. U tim tablicama pohranjuju se strukture koncepata i konceptualnih mapa koje je korisnik stvorio u procesu učenja. U tablici `koncept` bilježe se podaci svakog koncepta koji korisnik

evidentira. Uz svaki koncept može se vezati više resursa. Tablica `mapa` služi za pohranu konceptualnih mapa i u njoj je između ostalih podataka pohranjena kompletna struktura mape u CXL formatu. Tablica `kon_mapa` služi za povezivanje koncepata s konceptualnim mapama, jer se jedan koncept može nalaziti u više mapa. Mape koje su označene kao javne, dostupne su za zajednički rad svim korisnicima sustava, a jedino što je dozvoljeno isključivo autoru je brisanje mape.

Tablica `korisnik_poruka` služi kao poštanski sandučić pri razmjeni korisničkih poruka. Princip razmjene je identičan razmjeni poruka kod klasičnih e-mail poruka, a svaki korisnik ima uvid u primljene i poslane poruke.

Podaci o sinkronoj komunikaciji smješteni su u tablicama `chat_tema`, `chat_poruka`, `chat_sudionici` i `chat_cekaj`. Tablice `chat_tema`, `chat_poruka` i `chat_sudionici` koriste se za bilježenje podataka o tijeku komunikacije, temama koje imaju ulogu identičnu prostorijama za čavrljanje u klasičnoj komunikaciji čavrljanjem (engl. *chat room*) i sudionicima komunikacije. Četvrta tablica `chat_cekaj` ima ulogu binarnog semafora koji se postavlja na odgovarajuću vrijednost kada se u komunikaciji o određenoj temi dogodi neka promjena (npr. korisnik pošalje novu poruku ili se u komunikaciju priključi novi korisnik). Ta tablica je potrebna kao komunikacijski mehanizam jer poslužiteljska PHP tehnologija nema mogućnost samostalnog slanja podataka klijentima ako oni prethodno nisu uspostavili vezu.

6.5 Opis specifičnih rješenja

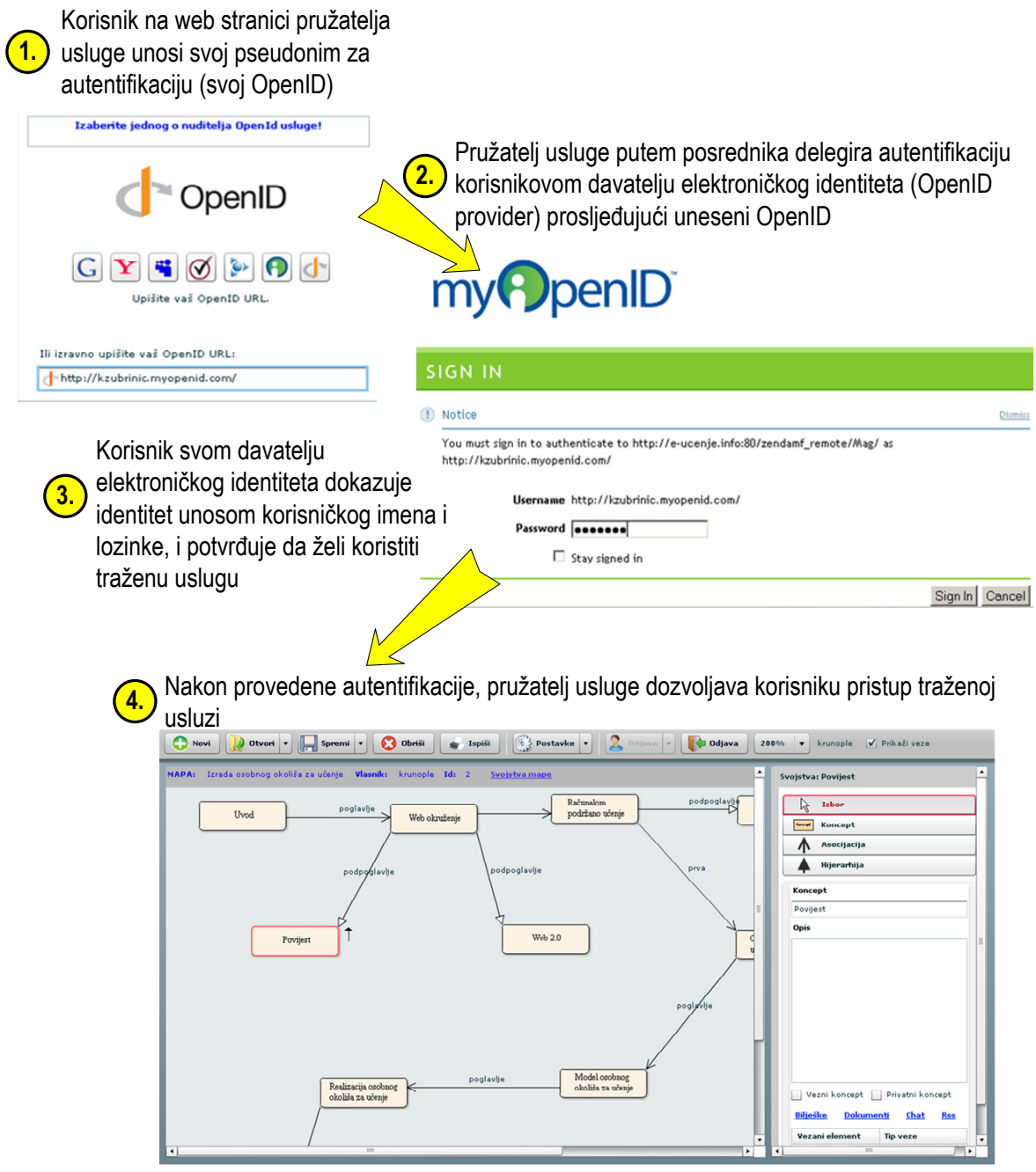
U ovom poglavlju opisan je način na koji su riješeni neki specifični problemi na koje se naišlo tijekom razvoja.

6.5.1 Utvrđivanje identiteta korisnika

Za provjeru identiteta korisnika izabran je *OpenID* protokol [109] zbog svoje jednostavnosti u implementaciji i široke baze korisnika.

Korisnik unosi svoju oznaku *OpenID* identiteta na početnoj stranici aplikacije. Pružatelj usluge iz unesene oznake saznaje tko je davatelj elektroničkog identiteta i preko posrednika prosljeđuje zahtjev za utvrđivanjem identiteta. Posrednik vrši preusmjeravanje s web stranice pružatelja usluga na web stranicu davatelja elektroničkog identiteta. Komunikacija se provodi korištenjem HTTPS protokola kako bi se smanjila mogućnost upada i krađe korisničkih podataka. Korisnik se na web stranici davatelja elektroničkog identiteta identificira navođenjem svoje lozinke i mora potvrditi da stvarno želi pristupiti usluzi s koje je preusmjeren. Ako je unesena lozinka ispravna, a korisnik je potvrdio da želi koristiti uslugu, davatelj elektroničkog identiteta vraća pružatelju usluge odgovor s rezultatima identifikacije. Pružatelj usluge na osnovu odgovora dozvoljava pristup ili odbija korisnika.

Na slici 6.8 prikazan je proces provjere identiteta korisnika pomoću *OpenID* protokola.



Slika 6.8 Proces provjere identiteta korištenjem *OpenID* protokola

6.5.1.1 Realizacija procesa provjere identiteta

Pošto se tijekom provjere identiteta pomoću *OpenID* protokola koristi preusmjeravanje korisnika između dvije web stranice (aplikacije i web stranice davatelja *OpenID* identiteta), bilo je nužno na neki način sačuvati stanje aplikacije u trenutku započinjanja procesa kako bi se po povratku korisnik našao na istom mjestu odakle je pošao. Nastali problem proizlazi iz karakteristika izabranog okruženja, a to je što *Flash player* radi s vrlo malim podskupom HTML-a, pa nije bilo moguće izravno umetnuti web stranicu unutar *Flash* aplikacije. Zbog toga je izabran pristup koji je za provjeru identiteta korištenjem *OpenID* protokola u RIA

okruženju predložio Luke Shepard [70], a to jer da aplikacija otvori novi HTML prozor koji će se koristiti u tom procesu. U prvoj fazi provjere nakon što korisnik unese svoju oznaku *OpenID* identiteta, posrednik realiziran u obliku JS programa će otvoriti novi *pop-up* prozor i pozivom PHP skripte proslijediti zahtjev za utvrđivanjem identiteta odgovarajućem davatelju. Korisnik će se identificirati na njegovoj web stranici unutar otvorenog *pop-up* prozora. *OpenID* davatelj će vratiti rezultat provjere, a taj rezultat će dohvatiti JS funkcija realizirana korištenjem *YUI Ajax* biblioteke [125]. Korištena biblioteka omogućuje asinkronu komunikaciju i razmjenu podataka bez potrebe za osvježavanjem čitave web stranice prilikom prijema novih podataka. JS funkcija će po prijemu podatke proslijediti *Flash* aplikaciji koja će ovisno o rezultatima provjere identiteta nastaviti proces prijave korisnika u sustav ili će mu ispisati poruku o neuspješnoj provjeri identiteta. Po završetku procesa otvoreni *pop-up* prozor će se automatski zatvoriti. Ako iz bilo kojeg razloga unutar zadanog vremenskog intervala od deset sekundi nakon utvrđivanja identiteta ne stigne odgovor od davatelja *OpenID* identiteta, aplikacija će smatrati da se dogodila greška pa će prekinuti proces i zatvoriti *pop-up* prozor.

Preduvjet za korištenje realiziranog načina provjere identiteta je da se web stranici na kojoj se nalazi aplikacija aplikacije dozvoli otvaranje *pop-up* prozora i izvršavanje JS funkcija.

6.5.1.2 Forma za utvrđivanje identiteta

Način provjere identiteta korisnika pomoću *OpenID* protokola sve se više koristi na webu, ali korisnici su još uvijek slabo upoznati s tim pojmom, tako da je prva reakcija korisnika često izbjegavanje korištenja takvih usluga. Mnoge poznate Web 2.0 kompanije su se uključile kao davatelji koji omogućuju korištenje svojih korisničkih imena kao *OpenID* identiteta (npr. *Google*, *Yahoo* [URI42], *MySpace*).

Kako bi se olakšalo provođenje procesa prijave, napravljena je forma na kojoj su prikazani neki od najpopularnijih davatelja *OpenID* identiteta. Na ekranu se nalazi šest tipaka s logotipovima kompanija (*Google*, *Yahoo*, *MySpace*, *Verisign* [URI37], *AOL* [URI4] i *MyOpenID* [URI24]). Izravnim pritiskom na odgovarajuću ikonu pokreće se proces provjere identiteta korištenjem izabranog davatelja. Sedma ikona je predviđena za slobodan unos oznake bilo kojeg drugog davatelja *OpenID* identiteta.

Nakon pritiska na željenu tipku, aplikacija će iz unesenih korisnikovih podataka i informacije o web adresi OpenID davatelja usluge odrediti URI na kojem se nalazi forma za provjeru identiteta korisnika, te će preusmjeriti korisnika na tu web stranicu. Na toj stranici korisnik mora unosom svojih korisničkih podataka dokazati svoj identitet, a nakon toga potvrditi da želi pristupiti traženoj usluzi.

Klijentska aplikacija će zapamtiti posljednjeg korištenog *OpenID* davatelja usluga te će ga ponuditi korisniku kao prvi izbor prilikom slijedećeg korištenja. Jedan korisnik može koristiti više različitih *OpenID* pseudonima kako bi se smanjila mogućnost praćenja aktivnosti korisnika, i omogućio rad u slučaju da web stranica određenog *OpenID* davatelja usluga prilikom prijave nije u funkciji. U tom slučaju, korisnik će za prijavu koristiti *OpenID* oznaku drugog davatelja usluga.

6.5.2 Postavke aplikacije

Svaki korisnik može samostalno podesiti postavke aplikacije koje se pohranjuju u bazu podataka na web poslužitelju. Početna ideja od koje se tijekom razvoja odustalo bila je da se postavke pohranjuju u XML datoteci smještenoj na poslužitelju u direktoriju svakog korisnika. Prilikom probnog korištenja aplikacije utvrđeno je da takav način pohrane podataka kod stvaranja i ažuriranja podataka uzrokuje usporenje rada aplikacije, pa je za spremište izabrana relacijska baza. Preostala je jedna XML datoteka koja se odnosi na sve korisnike aplikacije i u njoj su zapisani podaci potrebni za prijavu na relacijsku bazu podataka.

Za korištenje osnovnih funkcionalnosti servisa poput pretraživača najčešće nisu potrebni nikakvi autorizacijski podaci, iako određeni API-ji zahtijevaju da aplikacija prilikom korištenja šalje oznaku koja ju identificira. Takav pristup vlasnici servisa koriste kako bi mogli pratiti korištenje svojih resursa, te kako bi mogli lakše odrediti izvor problema u slučaju većeg korištenja resursa. Prilikom podešavanja web servisa koriste se dvije vrste postavki. Jedne su postavke aplikacije, a druge postavke konkretne osobe.

Postavke aplikacije su zapisane u bazi i koriste ih svi korisnici prilikom dohvata podataka. Npr. svi korisnici aplikacije pristupaju čitanju sadržaja s *YouTube* servisa koristeći istu oznaku i ključ aplikacije.

Kada je potrebno pristupiti privatnim podacima, pohraniti ili promijeniti neki sadržaj smješten na webu, osoba koja to radi mora se osobno identificirati web servisu. Npr. za pohranu bilješki korištenjem *Google* tabličnog kalkulatora korisnik se mora identificirati svojim *Googleovim* korisničkim imenom. Postavke te vrste korisnici mogu samostalno evidentirati unutar aplikacije.

Dio postavki trenutnog korisnika se pohranjuju lokalno na klijentskom računalu korištenjem tehnologije dijeljenih objekata (engl. *Shared Objects*). Dijeljeni objekti su tehnologija vezana uz *Flash* okruženje, a služe za lokalnu pohranu podataka. Način funkcioniranja vrlo je sličan načinu na koji funkcionira pohrana *cookieja* u web pregledniku. Prednost dijeljenih objekta je što nisu ograničeni samo na pohranu teksta već se pomoću njih mogu pohranjivati kompletni AS3 objekti. U aplikaciji se dijeljeni objekti koriste za privremenu pohranu korisničkih podataka. Nakon što se korisnik autorizira za rad s aplikacijom, njegovi osnovni podaci se pohranjuju na lokalno računalo. Prilikom svakog slijedećeg pristupa aplikacija će prvo provjeriti postoje li lokalno zapisani korisnički podaci unutar *Flash* okruženja i ako postoje učitati će ih, a korisnik neće morati ponavljati proces utvrđivanja identiteta i prijave. Ako korisnik pristupa aplikacija s računala koje dijeli s drugima, obavezno se po završetku rada u aplikaciji treba odjaviti pritiskom na tipku za odjavu kako bi se obrisali svi lokalno pohranjeni korisnički podaci.

6.5.3 Plan učenja

Konceptualna mapa se u klijentskom dijelu aplikacije koristi za prikaz strukture znanja i služi kao polazište iz kojeg se pristupa prikupljenim materijalima. Pri stvaranju mape korisnik iz izbornika aplikacije bira alat za crtanje odgovarajućeg elementa mape (čvora ili veze). Nakon

što je element kreiran, otvara se forma na kojoj mogu zabilježiti svojstva kreiranog elementa (naziv, opis i vrsta). Novi element se odmah uključuje u mapu pohranjenu u memoriji, a da bi bio trajno pohranjen na poslužitelju, korisnik mora izabrati opciju spremanja mape na poslužitelj. Klijent poziva funkciju servisa na poslužitelju koji mapu pohranjuje u bazu podataka. Za pohranu strukture i vizualnog izgleda konceptualne mape aplikacija koristi CXL format. Taj format je razvijen na IHMC institutu na Floridi u sklopu projekta razvoja CMAP alata za podršku izradi konceptualnih mapa, a koristi se kao dio sučelja za komunikaciju s CMAP poslužiteljem [11]. Format služi za pohranu i razmjenu podataka o konceptualnim mapama između različitih aplikacija. Baziran je na XML-u, a izabran iz razloga što omogućuje jednostavnu razmjenu konceptualnih mapa između različitih aplikacija, te pohranu na CMAP poslužitelj bez dodatne pretvorbe podataka. Osim toga, pomoću tog formata moguće je osim logičke strukture mape pohraniti i vizualne atribute.

6.5.3.1 Opis CXL formata

CXL datoteka sastoji se od 4 osnovna dijela. Prvi dio definira tip XML dokumenta i imeničko područje, a u drugom dijelu se navode informacije o konceptualnoj mapi (opis mape, podaci o autoru i sl.). Treći dio je središnji i u sebi nosi informacije o konceptima, njihove atribute i veze među njima. Posljednji, četvrti dio definira informacije važne za grafičku prezentaciju koncepata i veza (vrstu, boju i veličinu slova, boju pozadine, smještaj koncepata i veza i sl.). Podaci u drugom i četvrtom dijelu nisu obvezni, tako da je format moguće koristiti za opis logičke strukture mape neovisno o grafičkoj prezentaciji, ali i kao format koji osim logičke strukture podataka prenosi i informaciju o vizualnom izgledu mape.

Korijenska oznaka dokumenta unutar koje se navode svi ostali podaci je `cmap`, a njezini atributi definiraju korištena imenička područja. Unutar korijenskog elementa ugnježđeni su elementi `res-meta` i `map`. Element `res-meta` nije obavezan, a unutar njega se navode meta podaci o dokumentu. (naziv i opis mape te podaci o autorima).

Podaci o konceptima i vezama među njima navode se kao sadržaj elementa *map* koji sadrži dvije skupine informacija. Prva skupina obuhvaća podatke o logičkoj strukturi mape (konceptima i vezama među njima) koji se navode unutar oznaka `concept-list`, `linking-phrase-list` i `connection-list`. Element `concept-list` sadrži popis svih koncepata definiranih elementima `concept`. Obavezni atributi svakog koncepta su `id` koji predstavlja jedinstvenu oznaku koncepta unutar mape i `label` koji predstavlja tekst koji se ispisuje na oznaci koncepta. Koncepti su povezani veznim frazama koje se navode unutar elementa `linking-phrase-list`, dok su sve veze sadržaj elementa `connection-list`. Veze povezuju koncept i veznu frazu. Važni atributi svake veze su `id` koji predstavlja jedinstvenu oznaku veze unutar mape te `from-id` i `to-id` koji predstavljaju oznaku početnog i završnog koncepta, odnosno vezne fraze koje ta veza povezuje.

U četvrtom dijelu unutar elementa `concept-appearance-list` i `linking-phrase-appearance-list` navode se informacije važne za grafičko predstavljanje koncepata i veza (pozicije, debljina i boja linije, vrsta, veličina i boja slova, i sl.).

Osim četiri opisana dijela koji nose informacije o strukturi i vizualnom izgledu mape, specifikacija jezika predviđa i elemente koji definiraju opis sadržaja pridruženog određenim konceptima. Detaljna specifikacija CXL jezika s primjerima može se pronaći u dokumentaciji CMAP projekta [98].

Na slici 6.9 prikazan je primjer konceptualne mape opisane CXL jezikom.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--1. dio: tip XML dokumenta i imeničko područje -->
<cmap xmlns="http://cmap.ihmc.us/xml/cmap.dtd"
xmlns:dc="http://purl.org/dc/elements/1.1/">
      <!--2. dio: meta podaci o konceptualnoj mapi -->
      <res-meta>
            <dc:title>Biljke</dc:title>
            <dc:description>Opis biljke</dc:description>
      </res-meta>
      <map>
            <!-- 3. dio: podaci o konceptima -->
            <concept-list>
                  <concept id="1" label="Biljka"/>
                  <concept id="2" label="List"/>
                  <concept id="3" label="Korijen"/>
                  <concept id="4" label="Cvijet"/>
            </concept-list>
            <linking-phrase-list>
                  <linking-phrase id="5" label="ima"/>
            </linking-phrase-list>
            <connection-list>
                  <connection from-id="1" to-id="5"/>
                  <connection from-id="5" to-id="2"/>
                  <connection from-id="5" to-id="3"/>
                  <connection from-id="5" to-id="4"/>
            </connection-list>
            <!-- 4. dio: podaci o grafičkoj prezentaciji -->
            <concept-appearance-list>
                  <concept-appearance id="1" x="73" y="56"/>
                  <concept-appearance id="2" x="136" y="158"/>
            </concept-appearance-list>
            <linking-phrase-appearance-list>
                  <linking-phrase-appearance id="3" x="104" y="107"/>
            </linking-phrase-appearance-list>
      </map>
</cmap>
```

Slika 6.9 Konceptualna mapa opisana CXL jezikom

6.5.3.2 Korištenje CXL formata u aplikaciji

U aplikaciji *Samouk* koncepti unutar mape povezuju se izravno bez korištenja posebnih veznih fraza, a atributi se pridružuju vezi. Takav način povezivanja ne podržava višestruke veze koje imaju više ulaza i jedan izlaz (i obratno). Kako bi se CXL format mogao koristiti u aplikaciji, bilo je nužno napraviti određene prilagodbe u aplikaciji, te proširiti sam format.

Izvorni CXL format nema mogućnost pridruživanja atributa vezama, već su dva koncepta međusobno povezana s jednom veznom frazom i dvije veze. Zbog toga je za potrebe prve verzije aplikacije CXL format proširen na način da je elementu koji opisuje vezu (element `connection`) pored postojećih (`id`, `from-id` i `to-id`) atributa dozvoljeno pridruživanje

atributa `label` za prikaz naslova i `short-comment` za prikaz opisa veze. Na takav način atributi veze se mogu pohraniti u CXL format. Nedostatak takvog pristupa je u tome što će druge aplikacije koje budu učitavale podatke iz takve mape izgubiti informaciju o nazivu i opisu veze. U slijedećoj verziji aplikacije *Samouk* planira se u potpunosti podržati način na koji CXL format bilježi informacije o vezama, te će se time u potpunosti podržati razmjena podataka između različitih aplikacija.

CXL za povezivanje dva koncepta koristi vezne fraze čime se omogućuje stvaranje veza s više ulaza i izlaza. Takvu vezu je moguće realizirati pomoću više pojedinačnih veza, ali prikaz s jednom vezom koja ima više ulaza i izlaza obično je preglednij. Zbog toga je u aplikaciji *Samouk* uveden pojam *vezne fraze* na način da se određenom konceptu može postaviti vrijednost tog atributa. Takav objekt se semantički i vizualno razlikuje od običnog koncepta (vizualno se prikazuje drugačijim grafičkim simbolom - elipsom). Prilikom uvoza i izvoza podataka CXL formatom, vezne fraze se preslikavaju u tu vrstu objekata.

6.5.4 Realizacija čitača RSS *feedova*

Čitač RSS *feedova* dohvaća objavljeni sadržaj i prikazuje ga na zaslonu. Prilikom dohvaćanja sadržaja on se pohranjuje u bazi podataka. Razlog je taj što autori RSS *feedova* redovno osvježavaju svoj sadržaj i uklanjaju stare, a dodaju nove materijale. Posljedica toga je da korisnici uvijek imaju uvid u trenutno aktualne sadržaje ali nemaju u starije, a pristup starijim podacima je nekada vrlo važan. Rješenje za taj problem je stvaranje arhive RSS *feedova*. Kada se prvi korisnik aplikacije pretplati na neki sadržaj, podaci tog *feeda* će se zapisati u bazu. Svako osvježavanje *feeda* u tu će bazu dodavati nove zapise, dok se stari neće brisati. Na takav način svi korisnici će imati uvid u povijest tog *feeda*, ograničenu prvom prijavom prvog korisnika sustava na određeni *feed*. U sustav je moguće dodati redovito osvježavanje sadržaja na koje je barem netko od korisnika pretplaćen. Međutim postoji mnoštvo RSS sadržaja na koje u određenom trenutku nije pretplaćen nitko od korisnika, a željeti će im pristupiti u budućnosti.

Dva su moguća pristupa dohvaćanju starijih sadržaja *feedova*. Prvi pristup je optimističan i bazira se na pretpostavci da će svaki autor koji objavljuje RSS sadržaje održavati arhivu starih materijala u obliku arhivskog *feeda.* Pristupom tom sadržaju mogao bi se dobiti uvid u stare materijale. Ovo rješenje nije praktično jer ne postoji standardan način kojim se povezuju aktualni sadržaji s arhivskim pa to ovisi isključivo o korisniku. Neki korisnici neće uopće održavati arhivu i ovaj pristup za njih neće funkcionirati.

Drugi pristup se bazira na tome da se podaci *feedova*, ne čitaju izravno od korisnika već putem trećeg servisa koji služi kao arhiva. RSS čitači na webu su postali vrlo popularni posljednjih nekoliko godina, imaju veliku bazu korisnika i obično održavaju arhivu starih RSS sadržaja. Kada korisnik pristupa RSS *feedovima* putem njihovog sučelja ima pristup ne samo najnovijim sadržajima, već i čitavoj arhivi svih materijala koji su objavljeni u posljednjih nekoliko godina. *Google Reader* [URI16] jedan je od popularnih agregatora RSS *feedova* pomoću kojeg je moguće pristupiti arhivi starih sadržaja. U nekoliko navrata se najavljivalo da će *Google* pustiti u javni opticaj API za korištenje sadržaja svoje baze RSS

feedova, ali to još nije napravljeno. Korištenjem API-ja takvog web servisa moglo bi se pristupati RSS sadržajima neovisno o vremenu njihovog nastanka i ne bi bilo potrebno stvarati svoju arhivu.

6.5.5 Komunikacija korisnika

6.5.5.1 Asinkrona komunikacija

Asinkrona komunikacija je realizirana klasičnom razmjenom poruka poput elektroničke pošte. Svaki korisnik ima dva sandučića: jedan za primljene, a drugi za poslane poruke. Korisnik ima mogućnost uvida u poruke, slanja nove poruke i odgovora na neku od primljenih, a poruke se pohranjuju u bazi podataka na web poslužitelju.

Aplikacija bi se u budućnosti mogla unaprijediti na način da obavještava korisnika prilikom dolaska nove poruke. To bi se učinkovito moglo napraviti tako da poslužiteljska aplikacija po prijemu nove poruke pošalje informaciju klijentu, a klijentska strana bi korisnika mogla obavijestiti putem teksta na ekranu ili porukom u *pop-up* prozorčiću. Pošto tehnologija u kojoj je realiziran poslužitelj ne podržava način rada u kojem poslužitelj inicira aktivnost, čitanje tih podataka obavilo bi se prozivanjem poslužitelja od strane klijenta u redovnim intervalima (npr. svakih deset minuta). Učestalost prozivanja korisnik bi mogao podesiti u svojim korisničkim postavkama.

6.5.5.2 Sinkrona komunikacija

Kod sinkronog oblika komunikacije svi sudionici su istovremeno prisutni i razmjenjuju poruke. Aplikacija za sinkronu komunikaciju realizirana je po principu aplikacije za komunikaciju čavrljanjem, a korisnici komuniciraju vezano uz temu određenu izabranim konceptom. Sinkrona komunikacija odvija se uz podršku relacijske baze u kojoj se pohranjuju poruke. Poruke se u bazi čuvaju ograničeno vrijeme (oko sat vremena), a nakon toga se automatski brišu.

Tijekom razvoja poslužiteljskog dijela aplikacije naišlo se na niz problema kojima je uzrok nemogućnost tzv. *server push* načina rada [44] PHP tehnologije. Kod takvog načina rada klijenti se registriraju kod poslužitelja, a nakon što se dogodi neka promjena koje je važna za njih, o nastaloj promjeni ih obavještava poslužitelj započinjući vezu. Pošto PHP nema mogućnost rada na takav način problem je trebalo riješiti simuliranjem takvog načina rada korištenjem tehnike dugog prozivanja (engl. *long pooling*). Kod te tehnike poslužitelj odgovara na upite klijenata samo ako se od posljednjeg prozivanja dogodila neka izmjena na podacima [76] ili je, u slučaju PHP-a protekao interval u kojem skripta mora završiti zadanu obradu. Programski je to riješeno korištenjem principa binarnog semafora. Kada se dogodi promjena u komunikaciji (nova poruka, prijava ili odjava korisnika), poslužitelj zapisuje informacije o promjeni i postavlja vrijednost semafora. U prvom ciklusu (otprilike svake 2 sekunde) svi korisnici postavljaju upit poslužitelju. Poslužitelj provjerava da li je semafor postavljen, i ako jest, pročitati će iz baze podatke o svim promjenama koje su se u međuvremenu dogodile i vratiti ih svim klijentima koji te informacije prethodno nisu dobili.

Ako se nije dogodila nikakva promjena, poslužitelj neće poslati odgovor klijentima već će periodički provjeravati da li se dogodila promjena. Nakon što protekne zadano vrijeme, poslužitelj će poslati poruku klijentima da se nije dogodila nikakva promjena. Klijenti po prijemu svake poruke šalju nove upite poslužitelju.

Veza između klijenta i poslužitelja ostaje aktivna sve dok se korisnik nalazi na komunikacijskoj formi. Činjenica da veza između klijenta i poslužitelja mora ostati aktivna dulje vrijeme može dovesti do slabijih osobina rada sustava tako da bi za asinkronu komunikaciju bolje rješenje bilo korištenje poslužiteljske tehnologije koja ima mogućnost iniciranja veze.

7 ANALIZA KORIŠTENJA APLIKACIJE ZA PODRŠKU STVARANJU OSOBNOG OKOLIŠA ZA UČENJE

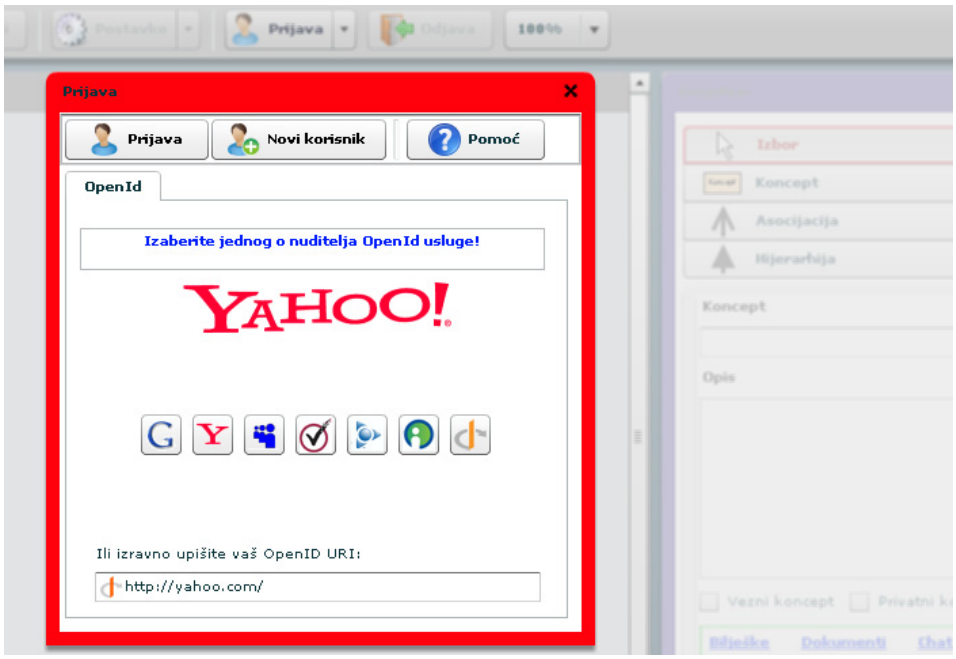
U ovom poglavlju analiziraju se mogućnosti korištenja aplikacije *Samouk* u procesu samostalnog učenja. U prvom dijelu opisano je korištenje sustava kroz nekoliko osnovnih slučajeva korištenja: utvrđivanje identiteta korisnika sustava, podešavanje sustava korisnikovim potrebama i proces učenja. Proces učenja promotren je na dva primjera. Na kraju se analiziraju rezultati provedenog testiranja i daje osvrt na daljnje unaprjeđenje i razvoj modela i aplikacije.

7.1 Osnovni slučajevi korištenja

7.1.1 Prijava u aplikaciju

Utvrđivanje identiteta korisnika provodi se unosom njegovog *OpenID* identiteta na početnoj stranici aplikacije. Korisnik ima na raspolaganju nekoliko inicijalno ponuđenih davatelja identiteta koje može izabrati klikom na odgovarajuću tipku. Ako davatelj identiteta nije među ponuđenima, novi *OpenID* se može izravno unijeti u predviđeno polje.

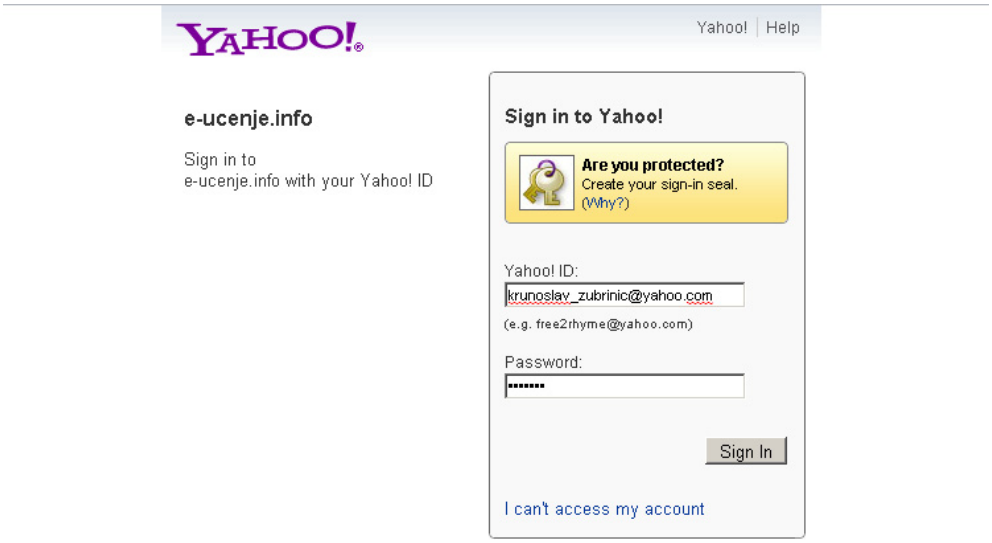
Na slici 7.1 prikazan je izgled ekrana za prijavu korisnika u aplikaciju. Boja okvira forme je crvena, što označava da korisnik još nije prijavljen.



Slika 7.1 Prijava u aplikaciju *Samouk*

Po započinjanju prijave u *pop-up* prozoru otvoriti će se web stranica izabranog davatelja *OpenID* identiteta na kojoj se korisnik mora prijaviti i potvrditi da stvarno želi pristupiti aplikaciji.

Ako je korisnik izabrao *Yahoo* kao davatelja usluge *OpenID* identiteta, izgled ekrana provjere identiteta prikazan je na slici 7.2.



Slika 7.2 Provjera identiteta korisnika aplikacije *Samouk* korištenjem *Yahooa* kao davatelja *OpenID* identiteta

Po uspješnoj prijavi pop-up prozor će se zatvoriti, a boja okvira forme za prijavu korisnika promijeniti će se u žutu što znači da je utvrđen identitet korisnika. Nakon toga program provjerava da li je korisnik evidentiran u aplikaciji. Ako je evidentiran, boja okvira forme će se promijeniti u zelenu, a korisnik će moći započeti s korištenjem aplikacije.

Ako se korisnik po završetku rada s aplikacijom ne odjavi, pri svakom slijedećem pristupu početnoj stranici automatski će se učitati korisničke postavke posljednjeg prijavljenog korisnika, i on neće morati prolaziti kroz proces prijave.

7.1.2 Administracija sustava

7.1.2.1 Prijava novog korisnika

Ako neki *OpenID* nije evidentiran u sustavu, korisnik se treba registrirati izborom funkcije unosa novog korisnika s ekrana prijave ili iz glavnog izbornika aplikacije. Prilikom registracije korisnik mora unijeti korisničke podatke od kojih su obavezni korisničko ime i barem jedna *OpenID* oznaka. Prilikom pridruživanja svakog *OpenIDja*, korisnik mora potvrditi da je on njegov vlasnik i proći kroz proces utvrđivanja identiteta opisan u prethodnoj točci. Nakon unosa korisničkih podataka, novi korisnik će odmah moći raditi s aplikacijom.

7.1.2.2 Podešavanje postavki web servisa

Da bi korisnik mogao koristiti web servise za dohvat i pohranu podataka tijekom procesa učenja, mora podesiti njihove postavke. Podešavanje postavki vrši se na kartici `Korisnički servisi` ekrana korisničkih postavki. Dvije skupine servisa imaju mogućnost podešavanja i to su `servisi za pohranu bilješki` i `servisi za`

`dohvat materijala`. Pri podešavanju, korisnik treba izabrati servise, unijeti korisničke postavke i potvrditi izbor servisa koje želi koristiti.

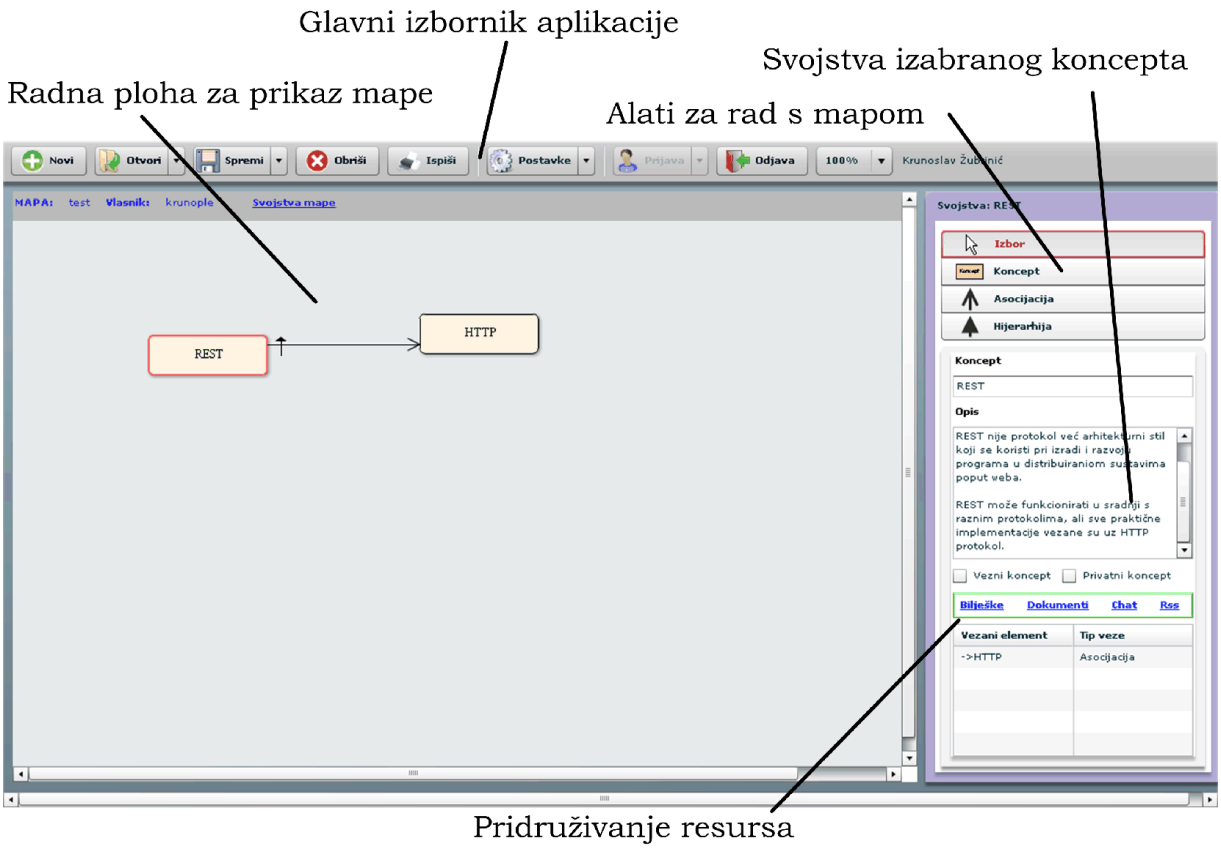
7.1.3 Proces učenja

U ovom dijelu opisati će se osnovni elementi procesa učenja koje aplikacija *Samouk* omogućuje: stvaranje plana učenja, pretraživanje materijala i njihovo pridruživanja dijelovima plana, izrada vlastitih materijala te komunikacija s drugim korisnicima sustava.

7.1.3.1 Opis korisničkog sučelja

Korisničko sučelje aplikacije *Samouk* sastoji se od tri osnovna dijela. Najveći dio zaslona zauzima radna ploha za prikaz plana učenja u obliku konceptualne mape. Na vrhu radne plohe prikazuju se naziv mape i ime autora, a omogućena je promjena postavki mape. U stupcu s desne strane smješteni su alati za rad s konceptualnom mapom, a na vrhu se nalazi glavni izbornik aplikacije. Izbornik s alatima za stvaranje i povezivanje koncepata nalazi se na vrhu desnog stupca, a ispod njega je smješten dio u kojem se prikazuju svojstva izabranog koncepta i preko kojeg se pristupa resursima vezanima uz taj koncept.

Izgled korisničkog sučelja aplikacije vidljiv je na slici 7.3.



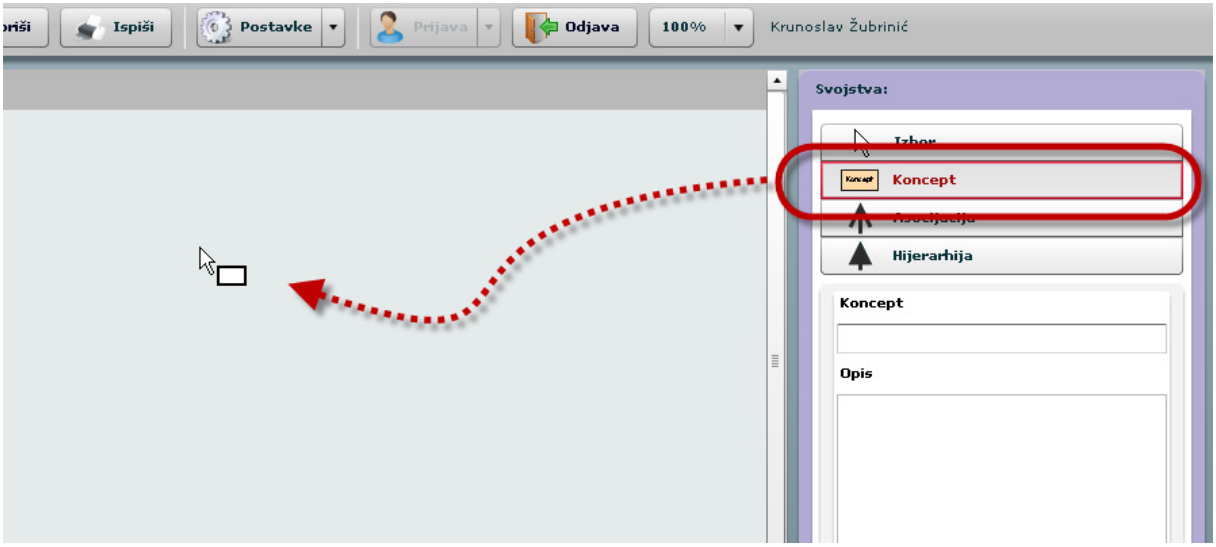
Slika 7.3 Korisničko sučelje aplikacije *Samouk*

U glavnom izborniku aplikacije nalaze se osnovne naredbe za rad s aplikacijom i mapom: stvaranje nove mape (izbornik `Novi`), otvaranje i uvoz postojećih mapa (izbornik `Otvori`), spremanje i izvoz mapa (izbornik `Spremi`), brisanje mape (izbornik `Obriši`), ispis

(izbornik `Ispiši`), podešavanje korisničkih postavki (izbornik `Postavke`) te prijava (izbornik `Prijava`) i odjava (izbornik `Odjava`) korisnika. Na alatnoj traci je tijekom rada vidljivo ime prijavljenog korisnika.

7.1.3.2 Stvaranje plana učenja

Izrada konceptualne mape koja predstavlja plan učenja započinje stvaranjem koncepata i njihovim međusobnim povezivanjem. Koncepti se stvaraju tako da korisnik izabere alat za stvaranje koncepta (tipka `Koncept`) iz izbornika smještenog na vrhu desnog stupca i nakon toga klikne na radnoj plohi na poziciju na koju želi smjestiti novi koncept, kako je prikazano na slici 7.4.

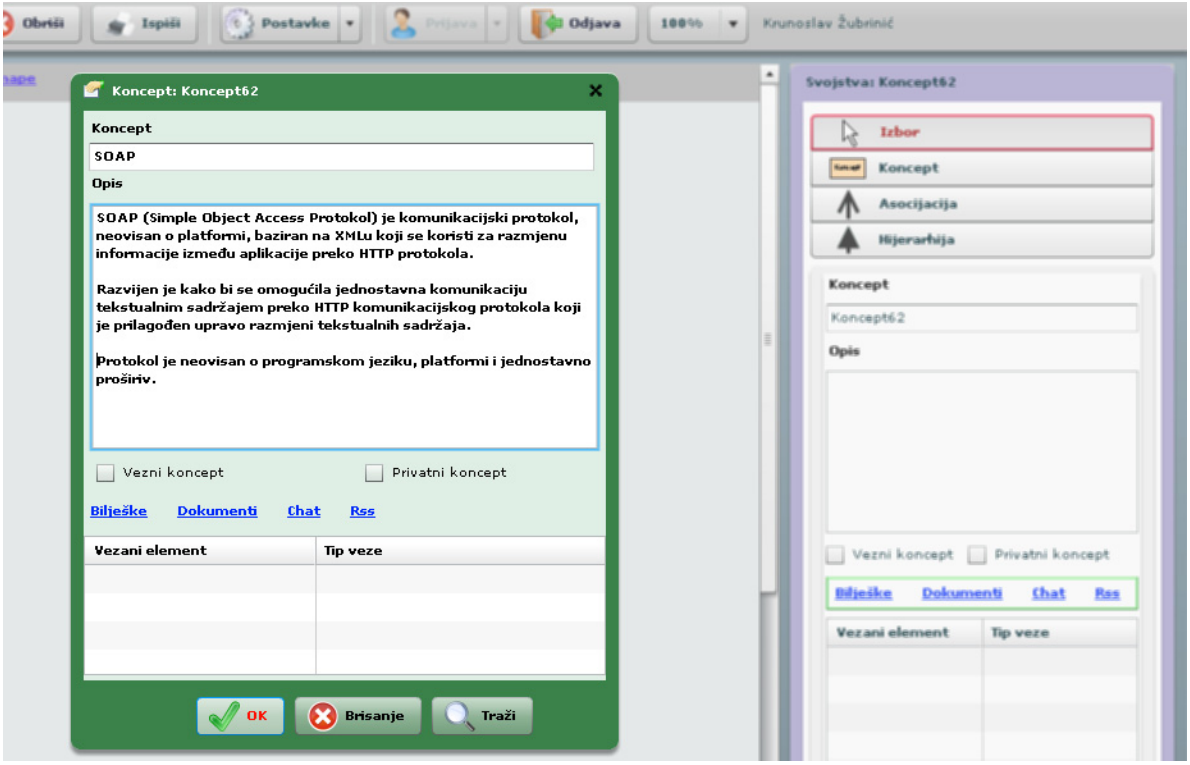


Slika 7.4 Stvaranje koncepta

Nakon što otpusti tipku miša, na plohi će se prikazati pravokutnik koji predstavlja koncept i automatski će se otvoriti forma za unos svojstava stvorenog koncepta.

Inicijalni naziv koncepta naveden ju naslovnoj traci forme. U polja forme unosi se naziv, opis, vrsta koncepta i oznaka koja označava da li je koncept privatan ili nije. Uneseni podaci pohranjuju se pritiskom na tipku `OK` forme. Pritiskom na tipku `Brisanje` stvoreni koncept se može obrisati i tada se brišu svi pridruženi resursi. Na istoj formi moguće je kopiranje opisa i vezanih materijala iz nekog drugog javno dostupnog koncepta. Pritiskom na tipku `Traži` otvara se forma na kojoj se prikazuju svi koncepti čija svojstva i materijali se mogu kopirati. Funkcionalnost preuzimanja svojstava iz drugog koncepta trenutno je vrlo ograničena i omogućuje preuzimanje sadržaja samo istoimenih koncepata.

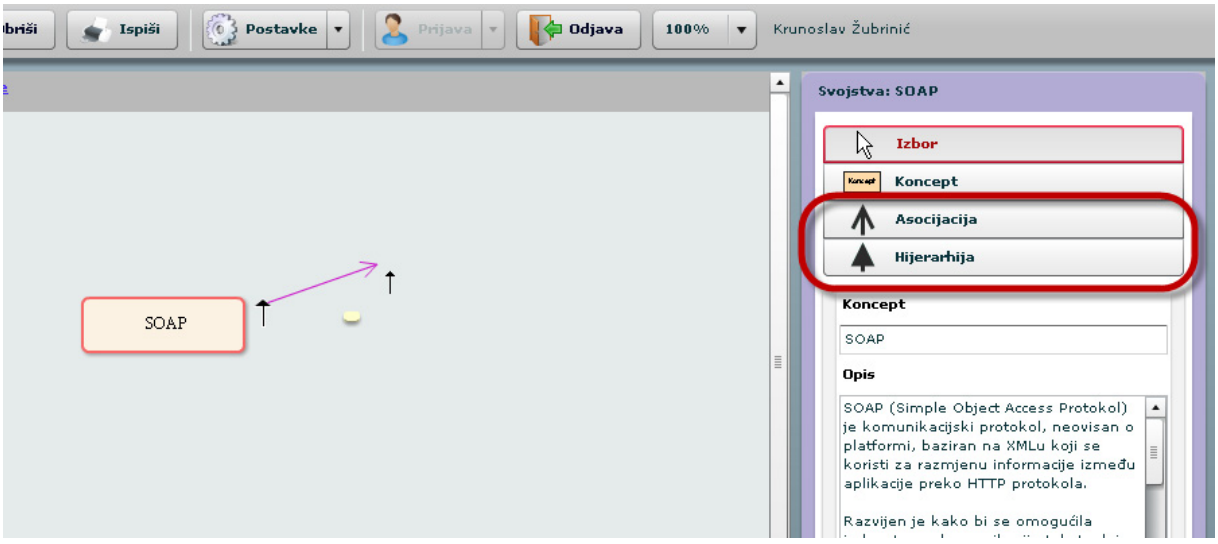
Na slici 7.5 prikazana je forma sa svojstvima koncepta.



Slika 7.5 Pridruživanje svojstava konceptu

Prilikom izrade mape podržano je stvaranje dvije vrste veza: asocijacije i hijerarhijskog uređenja. Novi koncept vezan uz prethodno kreirani korisnik može stvoriti tako da u izborniku s desne strane klikom izabere željenu vrstu veze (klikom na tipku `Asocijacija` ili `Hijerarhija`). Nakon toga lijevom tipkom miša treba kliknuti na polazni koncept i ne puštajući tipku, pokazivač odvući do mjesta na koje želi smjestiti novi koncept. Nakon što korisnik otpusti tipku miša, na poziciji pokazivača nastati će novi koncept.

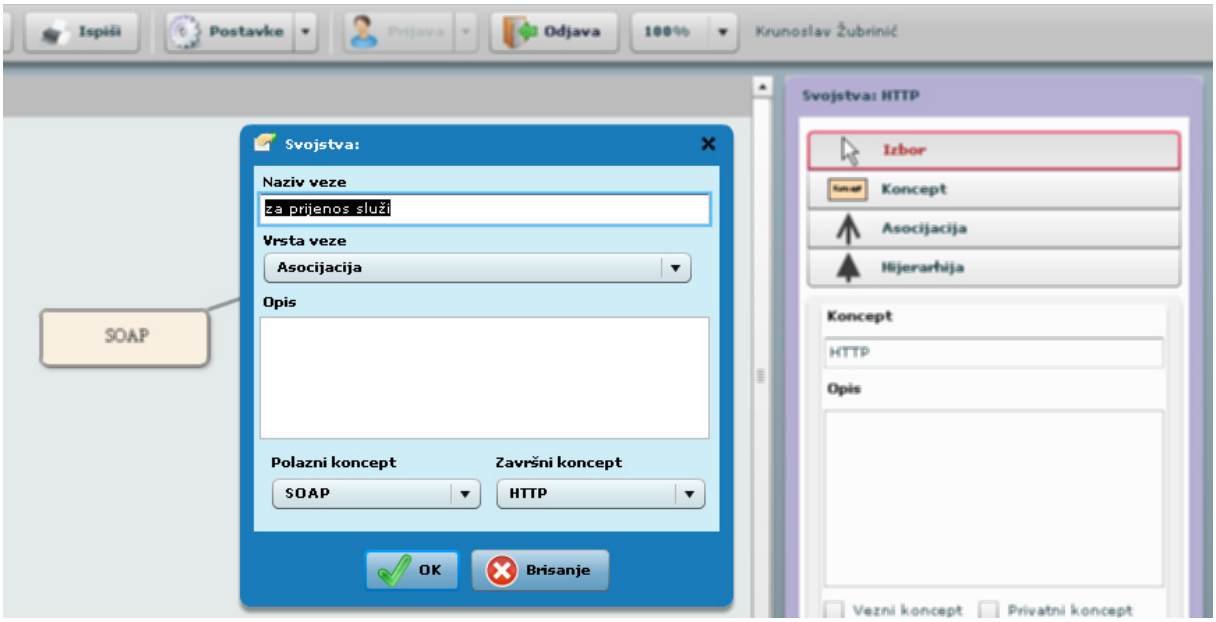
Stvaranje vezanog koncepta prikazano je na slici 7.6.



Slika 7.6 Stvaranje vezanog koncepta

Na isti način, odgovarajućom vrstom veze moguće je povezati dva prethodno kreirana koncepta. Inicijalno stvorena veza nema ime, a ako je potrebno, ime se može dodati dvoklikom na stvorenu vezu. Nakon toga će se otvoriti forma za evidenciju svojstava veza u kojoj je moguće unijeti naziv i opis veze. Zapisivanje svojstava veze radi se pritiskom na tipku `OK` forme. Na istoj formi moguće je obrisati vezu (pritiskom na tipku `Brisanje`), promijeniti vrstu veze (izborom željene vrste veze iz padajućeg izbornika `Vrsta veze`) te preusmjeriti vezu promjenom polaznog ili dolaznog koncepta (padajući izbornici `Polazni koncept` i `Dolazni koncept`).

Izgled ekrana prilikom promjene svojstva veze između koncepata prikazan je na slici 7.7.



Slika 7.7 Pridruživanje svojstva vezi između koncepata

Kreirani podaci konceptualne mape pohranjuju se u memoriji, a da bi ostali sačuvani, potrebno ih je spremiti na poslužitelj što se radi pritiskom na tipku `Spremi` glavnog izbornika.

7.1.3.3 Pridruživanje materijala

Svakom kreiranom konceptu mogu se pridružiti četiri različite vrste resursa: `dokumenti`, `bilješke`, `RSS sadržaji` i `tijek sinkrone komunikacije`. Podaci izabranog koncepta prikazani su u desnom stupcu u kojem se nalaze linkovi za pridruživanje resursa.

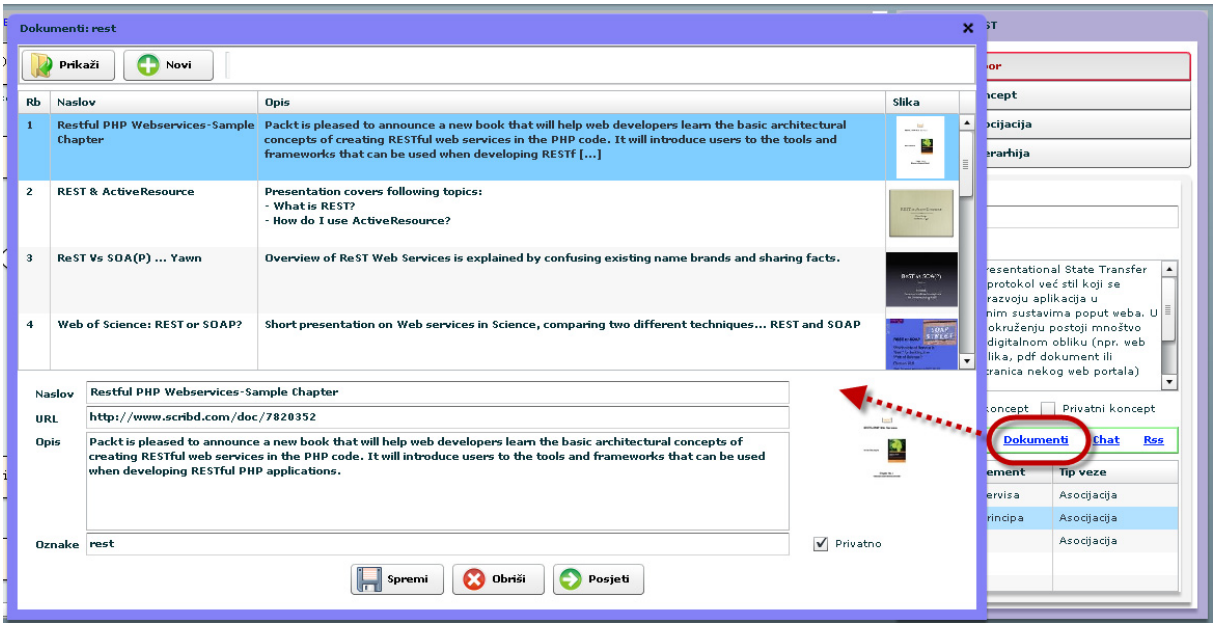
Klikom na vezu koja predstavlja izabranu vrstu resursa otvara se forma u kojoj su svi resursi koji su pridruženi izabranom konceptu. Na toj formi korisnik može pogledati sadržaj izabranog resursa (tipka `Posjeti`), dodati novi resurs (tipka `Novi`) ili obrisati izabrani resurs (tipka `Obriši`).

Elementi desnog stupca koji se koriste pri pridruživanju resursa vidljivi su na slici 7.8.

Slika 7.8 Pridruživanje resursa konceptu

Kod dodavanja novog resursa, pri pretraživanju je potrebno unijeti odgovarajući pojam i izabrati servise koji će se koristiti pri pretraživanju. Korisnik bira želi li pretraživati podatke korištenjem univerzalnih web pretraživača poput *Googlea* ili *Yahooa* ili specijaliziranih poput *Flickra*, *YouTubea* i sl. Tom prilikom može izabrati istovremeno pretraživanje pomoću više pretraživača. Rezultati pretraživanja prikazuju se u tablici. Sadržaje koji ga zadovoljavaju korisnik može pohraniti pritiskom na tipku `Spremi` te im pridružiti korisničke oznake. Pridruženi materijali su vizualno označeni ikonom kvačice i vidljivi su na formi za pregled pridruženih materijala. Na istom ekranu moguće je izravno, bez pretraživanja upisati podatke novog sadržaja, ili pohraniti datoteku na neki od web servisa podešenih za tu namjenu.

Po izvršenom pridruživanju materijala korisnik u tabličnom obliku ima uvid u sve materijale koji su pridruženi izabranom konceptu. Izgled toga ekrana vidljiv je na slici 7.9.

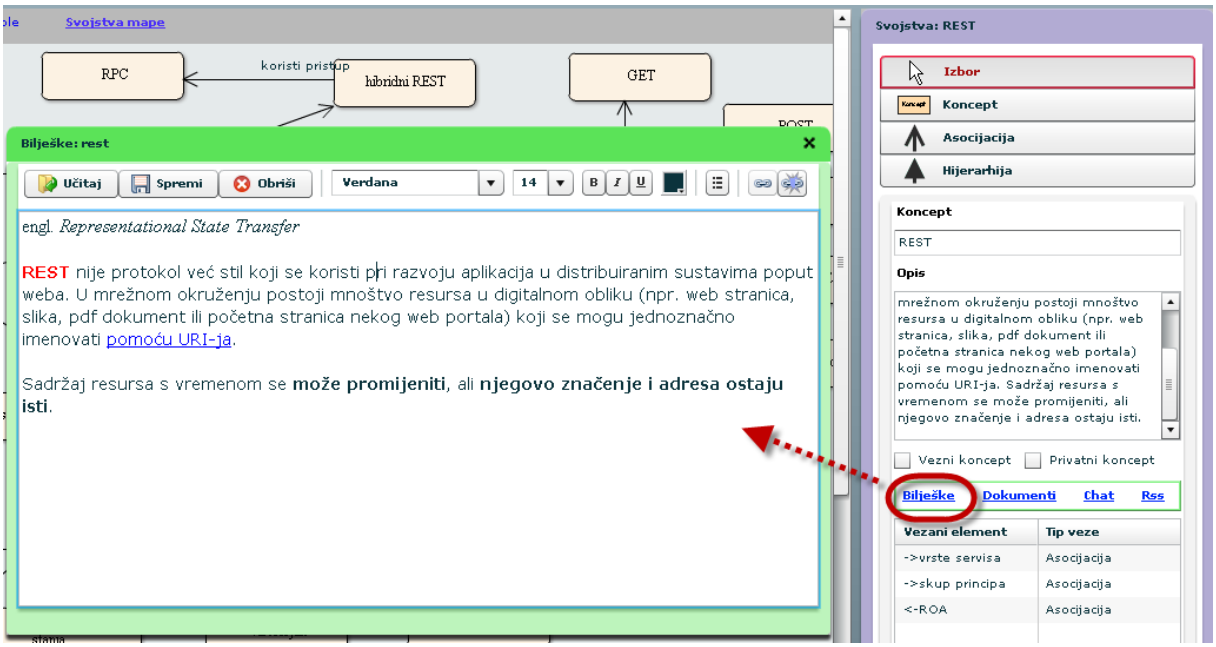


Slika 7.9 Materijali vezani uz koncept

Druga vrsta resursa koja se može pridružiti izabranom konceptu su bilješke. Izborom veze `Bilješke` pristupa se formi s vizualnim uređivačem teksta koji ima mogućnost osnovnog oblikovanja teksta: promjene vrste, veličine i boje slova, naglašavanja teksta (podebljan, nakošen ili podcrtan tekst), izrade liste i umetanja hiperveza. Izrađena bilješka pohranjuje se na web servis, a u postojećoj verziji aplikacije uz koncept je moguće vezati jednu bilješku.

Prilikom pohranjivanja, nestandardni HTML kod koji stvara *Flashov* vizualni uređivač teksta prevodi se automatski u standardni HTML 4 kod kako bi bio čitljiv na webu korištenjem drugih aplikacija. Prilikom dohvaćanja bilješke s web servisa, vrši se suprotna pretvorba, iz standardnog HTML 4 formata u nestandardni *Flashov*.

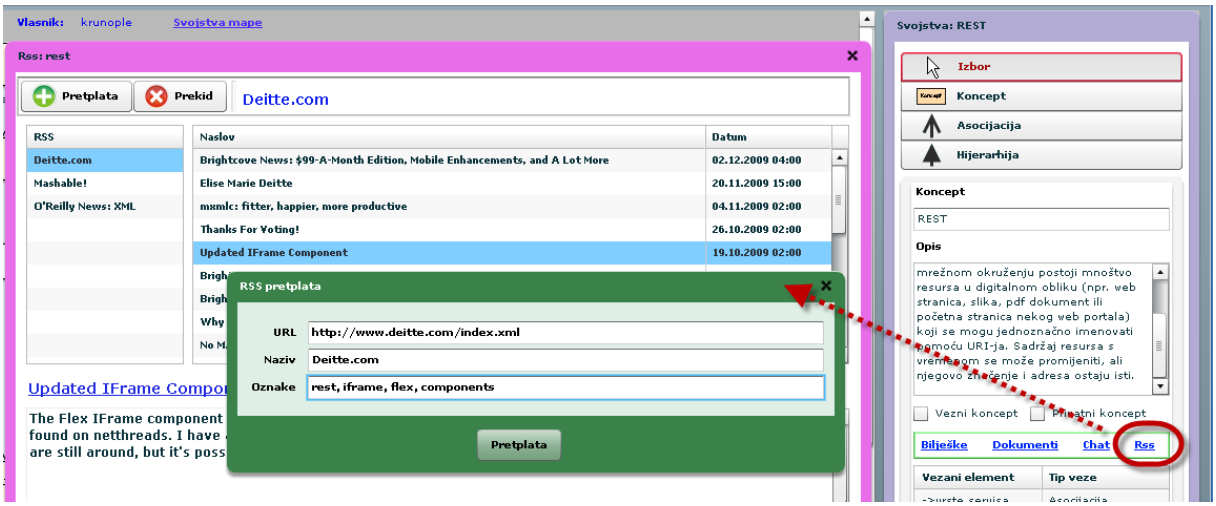
Forma za uređivanje bilješki prikazana je na slici 7.10.



Slika 7.10 Uređivanje bilješki

RSS sadržaji su treća vrsta resursa koji se mogu pridružiti uz izabrani koncept. Formi za uvid u RSS sadržaje pristupa se izborom veze `RSS`. Pri pretplati na novi sadržaj potrebno je unijeti njegov URI i korisničke oznake kojim ga želite označiti. Naziv nije potrebno unijeti osim ako se sadržaj želi nazvati nekim specifičnim imenom. Ako se naziv ne upiše program će prilikom pretplate pročitati naslov iz RSS *feeda*.

Na slici 7.11 prikazan je izgled ekrana prilikom pretplate na RSS sadržaj.

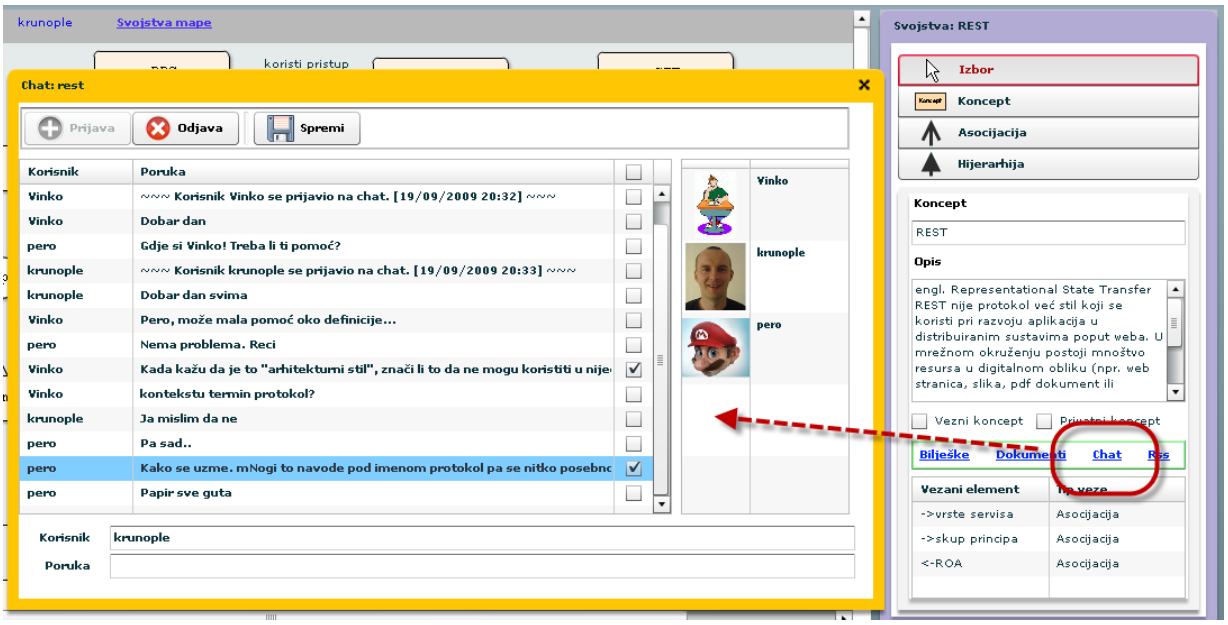


Slika 7.11 Pretplata na RSS sadržaj

7.1.3.4 Komunikacija s drugim učenicima

Komunikacija s drugim učenicima može se odvijati asinkrono i sinkrono. Asinkrona razmjena poruka funkcionira na način vrlo sličan razmjeni poruka elektroničke pošte. Korisnik ima mogućnost slanja i primanja poruka od drugih korisnika. Primljene poruke završavaju u dolaznom, a poslane u odlaznom sandučiću. Korisnik ima mogućnost pristupa porukama, njihovog brisanja i izravnog odgovora na primljenu poruku.

Sinkrona komunikacija odvija se u obliku komunikacije čavrljanjem, a korisnici komuniciraju vezano uz određenu temu predstavljenu izabranim konceptom. Izborom veze `Chat` korisnici ulaze u ekran za sinkronu komunikaciju prikazan na slici 7.12.



Slika 7.12 Tijek sinkrone komunikacije

Sudionici komunikacije mogu biti pasivni ili aktivni. Pasivni sudionici prate tijek komunikacije, ali u njoj ne sudjeluju aktivno. Korisnici koji žele slati poruke moraju se prijaviti čime postaju aktivni sudionici. Po izvršenoj prijavi, njihovi podaci (korisničko ime i slika) će postati vidljivi na komunikacijskoj formi.

Korisnici mogu izabrane dijelove sinkrone komunikacije pohraniti tako da ih označe kvačicom i izaberu naredbu spremanja. Nakon te akcije otvoriti će se forma s bilješkom vezanom uz koncept o kojem se vrši komunikacija, i u njezin sadržaj će se dodati izabrani dijelovi razgovora. Želi li trajno pohraniti te podatke, korisnik mora izabrati naredbu *Spremi* na formi za uređivanje bilješki.

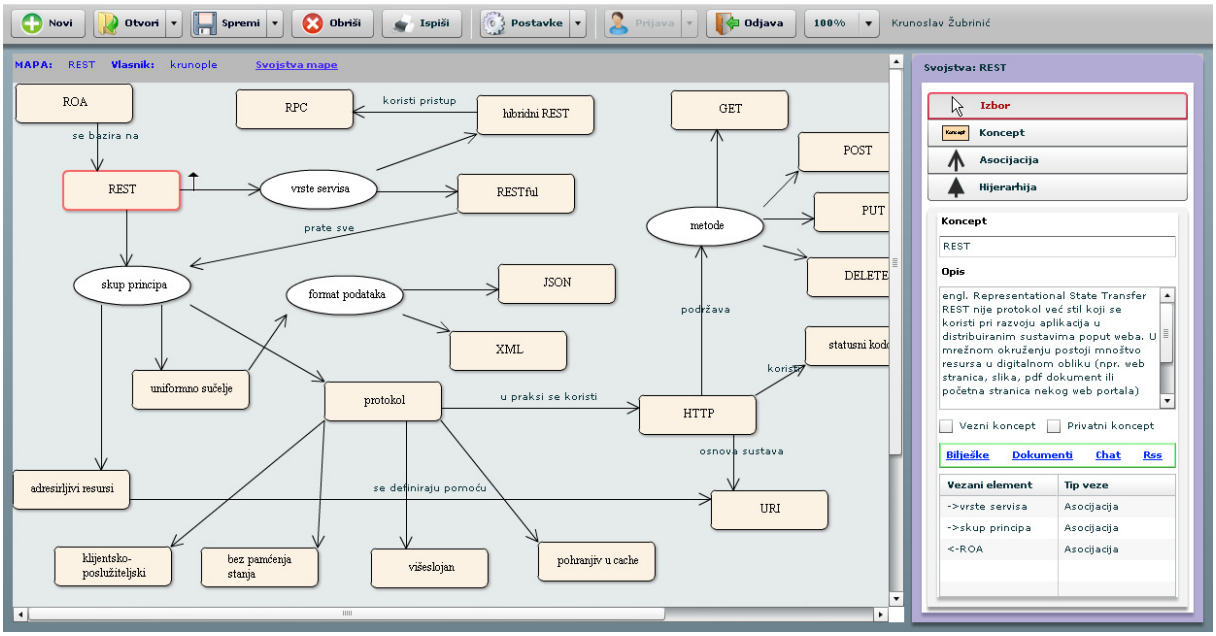
7.2 Primjeri korištenja aplikacije

U ovom dijelu opisati će se dva primjera korištenja aplikacije. Prvi je klasično samostalno učenje o izabranoj temi, a drugi se odnosi na korištenje aplikacije kao pomoćnog alata pri pisanju nekog strukturiranog teksta (npr. članka ili znanstvenog ili stručnog rada).

7.2.1 Primjer korištenja aplikacije u samostalnom učenju

Osnovna namjena aplikacije *Samouk* je potpora stvaranju osobnog okoliša koji se može koristiti prilikom samostalnog učenja. Kako bi se isprobala funkcionalnost za tu namjenu, izrađen je plan učenja o temi *REST pristup pri izradi web aplikacija*. Na početku je kreirana gruba mapa koja se sastojala od nekoliko osnovnih koncepata izravno vezanih uz REST princip: ROA, REST, HTTP, HTTP metode, URI, XML i JSON. Prikupljanjem materijala uz navedene koncepte slika je postajala složenija pa je početna, vrlo jednostavna mapa nadograđivana s novim konceptima.

Na slici 7.13 prikazan je izgled konačne mape koja prikazuje strukturu informacija o REST arhitekturnom pristupu.



Slika 7.13 Mapa koja predstavlja plan učenja

Koncepti su vezani vezama asocijacije, a zbog bolje preglednosti u nekim slučajevima korištene su imenovane veze s jednim ulazom i više izlaza. Korištene su jednosmjerne veze prikazane strelicom koja polazi iz jednog koncepta prema drugom, vezanom. U slučaju da je potrebno naglasiti da je veza između dva koncepta obostrana, potrebno je stvoriti još jednu vezu koja ide u suprotnom smjeru.

Osnovne bilješke o konceptu navedene su u polju opisa u svojstvima samog koncepta, a detaljnije su zabilježene na webu korištenjem web servisa za stvaranje bilješki. Materijali pronađeni na webu vezani su uz odgovarajuće koncepte.

Prednost ovakvog načina stvaranja plana učenja i mape znanja je u vizualnom prikazu. Uz svaki koncept mogu se pisati bilješke i pridruživati materijali dostupni na webu. Na takav način oni se vežu uz određeni pojam pa im se lako pristupa. Program omogućuje grupni rad više korisnika na zajedničkoj mapi čime se prikupljene informacije jednostavno nadograđuju.

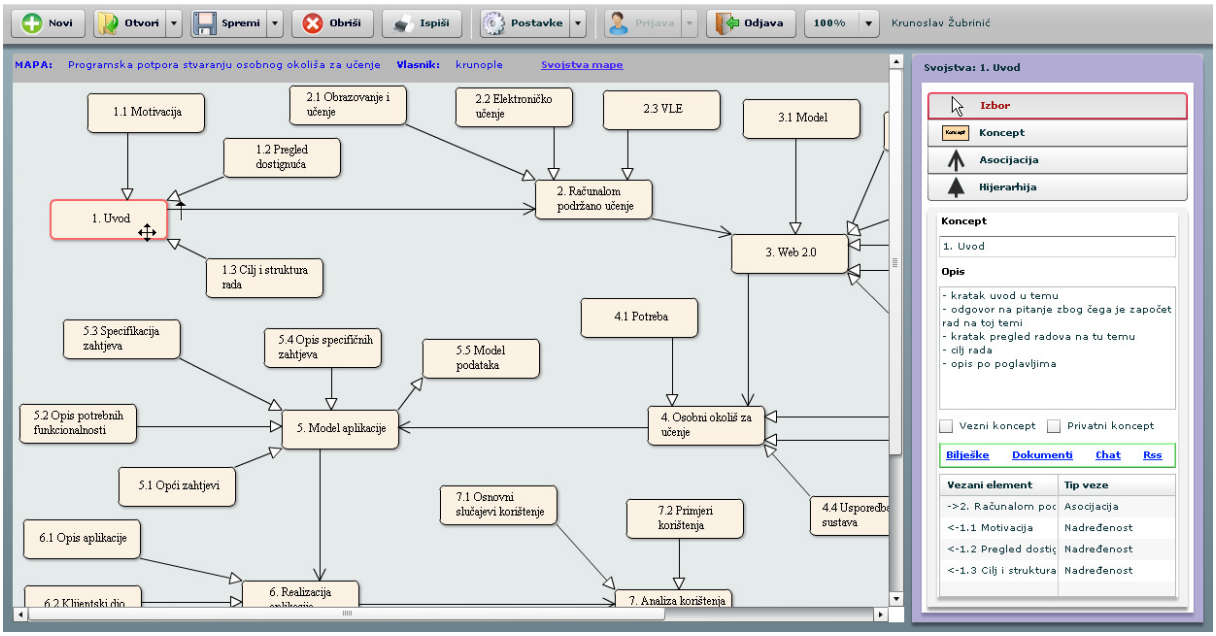
Nedostatak koji se javlja kod tema složenije strukture je što mapa može narasti i postati nepregledna za jednostavno korištenje. Jedan od načina na koji se taj problem može riješiti je izrada više manjih mapa i njihovo međusobno povezivanje.

7.2.2 Primjer korištenja aplikacije kao pomoćnog sredstva pri pisanju teksta

Okruženje stvoreno pomoću ove aplikacije može pomoći pri pisanju strukturiranog teksta (npr. znanstvenog ili stručnog rada, izvješća i sl.). Koncepti u tom slučaju predstavljaju osnovna poglavlja rada. Uz njih se hijerarhijskom vezom mogu vezati potpoglavlja do željene razine detalja. Sadržaj se može organizirati i na drugi način pa se uz osnovna poglavlja mogu vezati npr. elementi koji opisuju strukturu teksta (cilj, način pisanja, rokovi, literatura i sl.). Tekst samog rada evidentira se kao prilog odgovarajućem poglavlju, korištenjem servisa za izradu bilješki ili vezanjem URI-ja neke web aplikacije za uređivanje teksta (npr. *Google Docs ili Zoho Writer* [URI45]). Kod takvog načina rada dolazi do izražaja prednost korištenja aplikacija koje podržavaju provjeru identiteta korištenjem *OpenID* protokola. Ako se korisnik prijavi na ovu aplikaciju korištenjem *OpenID* identiteta, moći će izravno uređivati dokument unutar tog okruženja bez potrebe za ponovnom prijavom.

Hijerarhijska veza se prikazuje na način koji je običajan za prikazivanje generalizacije u UML dijagramima tako da strelica ide od specifičnog elementa prema općem. U slučaju razrade hijerarhije poglavlja, strelica pokazuje od potpoglavlja prema poglavlju. Prilikom testnog korištenja primijećeno je da je korisnicima koji su se prethodno upoznali s UML dijagramima značenje prikazane veze odmah jasno, dok drugi imaju problema sa shvaćanjem takvog prikaza. Prema njihovom mišljenju, prirodniji prikaz bi bio da strelica ide iz nadređenog elementa prema podređenima. Na osnovu korisničkih primjedbi tijekom daljnjeg testiranja sustava na većem broju korisnika, donijeti će se odluka o grafičkom prikazu hijerarhije.

Na slici 7.14 prikazana je mapa koja se koristi pri pisanju magistarskog rada. Elementi mape su poglavlja i potpoglavlja do druge razine.



Slika 7.14 Mapa koja se koristi pri pisanju rada

Vizualni prikaz strukture rada pomaže pri shvaćanju odnosa među poglavljima. Uz svako poglavlje rada mogu se pisati bilješke te vezati slike i linkovi na materijale u elektroničkom obliku tako da se resursi vežu uz ono poglavlje na koje se odnose. Program omogućuje grupni rad više korisnika na zajedničkom tekstu.

Nedostatak je što kod radova koji imaju složeniju strukturu, mapa može narasti i postati nepregledna za jednostavno korištenje. U trenutnoj verziji programa nije moguće sakriti određene detalje kreirane mape, već se prikazuju svi. Uvođenje mogućnosti skrivanja nepotrebnih detalja moglo bi pomoći jednostavnijem korištenju mape pri prikazivanju složenijih struktura. Funkcionalnost koja nije ugrađena u trenutnu verziju programa, a poželjna je za ovu namjenu, je ograničavanje korisnika koji mogu pristupati mapi na članove određene grupe.

7.3 Analiza korištenja aplikacije

Jedan od ciljeva magistarskog rada je izrada aplikacije za potporu stvaranju PLE-a. Prva verzija aplikacije *Samouk* s osnovnim funkcionalnostima je napravljena, instalirana na web poslužitelj i testirana u ograničenom okruženju koje nije reprezentativno, ali su dobivene korisne povratne informacije. Aplikaciju su testirale tri osobe koje su na početku ukratko upoznate s pojmovima PLE-a, konceptualne mape i osnovnim funkcionalnostima aplikacije. One su nakon toga aplikaciju koristile za stvaranje samostalnih mapa i prikupljanje materijala. Po obavljenom testiranju, usmeno su intervjuirane i zabilježena su njihova zapažanja i primjedbe na funkcionalnost aplikacije. Na osnovu povratnih informacija napravljena je analiza korištenja programa, zabilježeni su uočeni nedostaci i navedeni prijedlozi dorada potrebnih da bi se uočeni nedostaci otklonili.

7.3.1 Instalacija aplikacije

Poslužiteljska tehnologija u kojoj je aplikacija *Samouk* realizirana (PHP i relacijska baza *MySQL*) je česta na web poslužiteljima davatelja usluge smještaja web sadržaja što omogućuje instalaciju aplikacije na velikom broju web poslužitelja. Klijentski dio izrađen je u *Flash* tehnologiji i izvodi se u okruženju web preglednika čime se omogućuje korištenje aplikacije na svakom računalu koje ima instaliran web preglednik s *Flash player* dodatkom.

Instalacija aplikacije je jednostavna. Potrebno je provjeriti da li poslužitelj zadovoljava sve potrebe aplikacije (PHP verzije 5.1 ili viša i *MySQL* baza verzije 4 ili viša). Na poslužitelj je potrebno instalirati *Zend Framework* verzije 1.8.1 i *JanRain* PHP biblioteku verzije 2, a direktorije u koje su instalirane biblioteke potrebno je uključiti u PHP izvršnu putanju. Instalacija biblioteka svodi se na njihovo raspakiravanje u odgovarajućem direktoriju. Ako administrator instalira aplikaciju na web poslužitelj na kojem nema kontrolu nad sadržajem `php.ini` datoteke, putanju koja pokazuje na direktorije u kojima su instalirane korištene biblioteke mora postaviti kao vrijednost `$path` varijable u datoteci `ukljuci.php` smještenoj u korijenskom direktoriju PHP programa.

Na poslužitelju je potrebno napraviti dva direktorija. Prvi je direktorij u kojem će se nalaziti HTML stranica za izravan pristup klijentskom dijelu aplikacije. U taj direktorij potrebno je smjestiti sve datoteke klijentskog dijela aplikacije (HTML, SWF, CSS i JS datoteke). Drugi direktorij služi za smještaj PHP servisa. U taj direktorij potrebno je smjestiti sve datoteke poslužiteljskog dijela aplikacije.

Baza podataka se kreira izvođenjem sql skripte koja stvara strukturu svih potrebnih tablica i puni ih inicijalnim podacima. Potrebno je stvoriti jednog korisnika koji ima ovlasti za rad s kreiranom bazom. Dovoljno je da korisnik može pristupati podacima i ne mora imati administrativne ovlasti za promjenu strukture baze.

Da bi aplikacija radila na novom poslužitelju potrebno je napraviti izmjene u nekoliko datoteka. Prvo je u konfiguracijskoj datoteci *Flex* programskog okvira `services-config.xml` potrebno podesiti vrijednost atributa `uri` elementa `endpoint` s vrijednošću putanje koja vodi do direktorija u kojem su smješteni PHP servisi. Nakon toga je potrebno prevesti klijentski dio aplikacije kako bi se napravila nova izvršna *Flash* datoteka u koju je zapisana informaciju o adresi poslužitelja s kojim treba komunicirati AMF protokolom.

U datoteku `ple_ini.xml` koja se nalazi u korijenskom direktoriju PHP servisa potrebno je unijeti podatke potrebne za spajanje na bazu: naziv poslužitelja, naziv baze te ime i lozinku kreiranog korisnika baze koji će se koristiti za spajanje na bazu.

U datoteku `lokacijaIni.php` potrebno je unijeti putanju na poslužitelju koja vodi do datoteke `ple_ini.xml`, i URI koji pokazuje na lokaciju na kojoj su smješteni PHP servisi (isti podatak koji se unosi u `services-config.xml` datoteku).

Aplikaciji se pokreće pristupom početnoj HTML stranici klijentskog dijela.

7.3.2 Rezultati testiranja

U ovom dijelu navedene su primjedbe i prijedlozi korisnika prikupljeni tijekom testiranja aplikacije *Samouk*.

Način na koji je riješena prijava korisnika u sustav pomoću *OpenID* identiteta poznatih davatelja omogućuje jednostavnu prijavu i onim korisnicima koji nisu detaljno upoznati s pojmom *OpenIDja*. Prilikom same prijave korisnici često nisu obraćali pažnju na otvoreni *pop-up* prozor koji se koristi pri prijavi pa su smatrali da aplikacija ne radi. Nakon što su upozoreni na način na koji se prijava provodi, prijava je uglavnom prolazila bez problema. Zbog neuočljivosti bolji način provođenja prijave bio bi obavljanje prijave u okruženju same aplikacije, a ne u novom prozoru web preglednika.

Podešavanje korisničkih postavki trebalo bi doraditi tako da bude jednostavnije, npr. u obliku vodiča koji će korisnika voditi kroz sve postavke aplikacije koje treba podesiti. Novi korisnik prvo unosi osnovne korisničke podatke i nakon toga ga aplikacija prijavljuje u sustav. Međutim, aplikacija ga neće upozoriti u slučaju da nije izabrao web servise koje koristi unutar sustava. Primjedba je da bi svakom novom korisniku trebalo inicijalno pridružiti neke web servise.

Tekst sadržaja prikazuje se dosta sitnim slovima što nekim korisnicima stvara probleme u korištenju. U postavke koje korisnik može promijeniti svakako bi trebalo dodati mogućnost promjene stila prikaza elemenata (veličine, vrste i boje slova i sl.).

Način izrade mape je intuitivan i mapu je jednostavno stvoriti. Pri korištenju mape podaci se pohranjuju u memoriji sve dok korisnik ne izabere naredbu pohrane na disk. Jedna od primjedbi bila je da pri odlasku sa stranice nedostaje upozorenje u slučaju da sadržaj mape nije pohranjen. Način na koji program razmješta veze između koncepata je u nekim slučajevima nezgrapan i estetski ružan. Bolje bi bilo kada bi korisnici imali mogućnost samostalno odrediti pozicije veza između koncepata.

Svakom konceptu može se kroz svojstva pridružiti dulji tekst koji služi kao njegov opis. Istovremeno, konceptu se može pridružiti bilješka. Korisnike zbunjuju dva različita načina na koji se pridružuju podaci koji opisuju koncept. Primjedba je da je dovoljno jedno mjesto na kojem se pridružuje tekstualni opis konceptu. Osim tog, potrebno je omogućiti imenovanje i vezivanje više bilješki uz jedan koncept.

Korisnici nisu bili voljni podešavati web servise za pohranu bilješki. Primjedba je bila da bi inicijalno trebalo omogućiti pohranu bilješki unutar samog okruženja aplikacije bez korištenja web servisa. Na takav način neiskusan korisnik ne bi morao inicijalno podešavati postavke servisa koje su mu u prvi trenutak zbunjujuće. Naknadno bi korisnik mogao promijeniti način pohrane bilješki na pohranu u okruženju web servisa.

Način na koji se vrši povezivanje i pristup materijalima, ocijenjen je pomalo nespretnim jer ne omogućuje istovremeni uvid u sve materijale. Sadržaji u HTML obliku otvaraju se kod nekih web preglednika u novom prozoru, a kod drugih u novoj kartici istog prozora što je pri korištenju različitih web preglednika zbunjujuće za korisnika. Prijedlog za rješenje tog

problema isti je kao kod prijave korisnika, da se materijali otvaraju u okruženju aplikacije, a ne u novom prozoru web preglednika.

Autor nekog materijala nema utjecaj na promjenu sadržaja javno dostupnih materijala pošto ih slobodno mogu mijenjati svi korisnici. Na takav model data je primjedba da takvo nekontrolirano mijenjanje sadržaja nije dobro rješenje i da bi bolje bilo dozvoliti komentiranje javno dostupnih materijala, a ne i njihovo izravno mijenjanje.

Korisnici nisu u većoj mjeri koristili mogućnosti sinkrone i asinkrone komunikacije, a predloženi model su ocijenili zadovoljavajućim. Primjedba koju su dali je da bi se kompletna komunikacija trebala izdvojiti van okruženja određenog koncepta jer korisnici obično komuniciraju općenito, a rijetko vezano samo uz usko određenu temu. Rezultati općenite komunikacije trebali bi se moći pridruživati bilo kojem konceptu.

Generalni zaključak je da aplikacija *Samouk* može pomoći pri stvaranju PLE-a iako joj nedostaju određene funkcionalnosti potrebne za učinkovito korištenje, naročito pri učenju u grupama. Osim toga, trebalo bi poraditi na korisničkom sučelju kako bi ono bilo jednostavnije za korištenje i dopadljivije.

U nastavku je navedena detaljnija analiza uočenih problema i prijedlozi njihovog rješavanja.

7.3.3 Potrebne dorade po aplikaciji

7.3.3.1 Konceptualna mapa

Predstavljanje znanja konceptualnom mapom je korisno i pomaže lakšem shvaćanju cjelokupne slike, a povezivanje materijala uz koncepte pomaže kasnijem lakšem pronalaženju i pristupu materijalima.

Izrada složenijih mapa s više elemenata dovodi do smanjene preglednosti, čime se umanjuje korist koju donosi mapa. Jedan od načina na koji se taj problem može riješiti je izrada više manjih mapa koje prikazuju informacije iz užeg područja. Prilikom izrade novih mapa koje prikazuju složenije strukture, korisnici bi kao elemente mogli koristiti postojeće mape.

Drugi način na koji se složenost mape može smanjiti je dodavanje funkcionalnosti prikaza elemenata određene razine i skrivanja hijerarhijski podređenih elemenata. Npr. ako se sustav koristi kao pomoć pri pisanju teksta, a sadržaj se razrađuje po poglavljima do treće razine, korisnik u nekoj fazi izrade može izabrati prikaz elemenata koji pripadaju prvoj razini. U tom slučaju na zaslonu će biti vidljiva samo glavna poglavlja (*1. Uvod*, *2. Razrada*,…). Ako korisnik izabere prikaz elemenata druge razine vidjeti će sva poglavlja prve i druge razine (*1. Uvod*, *1.1 Motivacija*, *1.2 Dostignuća*, *1.3 Struktura*, *2. Razrada*,…). Da bi takav pristup funkcionalno zadovoljio potrebe korisnika, pri skrivanju elemenata niže razine trebali bi ostati dostupni svi resursi koji su vezani uz skrivene elemente. Npr. ako je uz element *1.2 Dostignuća* vezana slika koja prikazuje povijesni razvoj teme o kojoj je u radu riječ, ta slika bi trebala ostati dostupna preko nadređenog elementa *1.Uvod* u slučaju kada korisnik izabere prikaz elemenata prve razine.

Pozicije na kojima se prikazuju veze između koncepata određuje sama aplikacija, a oblik linije trenutno je ograničen samo na pravac. Korisniku bi trebalo omogućiti postavljanje linije na željenu poziciju i promjenu njezinog oblika (npr. prelom linije pod odgovarajućim kutom).

Svaki koncept i vezna fraza su jednake veličine tako da zauzimaju dosta prostora na zaslonu, a korisnik ne može utjecati na veličinu elemenata. Ergonomski bolji pristup bio bi inicijalno postavljanje veličine pravokutnika koncepata na veličinu teksta naziva. Nakon toga trebalo bi omogućiti korisniku promjenu veličine elemenata.

7.3.3.2 Podešavanje postavki sustava

Trebalo bi omogućiti podešavanje vizualnih postavki aplikacije od kojih je najvažnije promjena veličine i vrste slova koja se koriste u aplikaciji.

Aplikacija je rađena pod pretpostavkom korištenja na relativno velikim zaslonima, tako da prilikom korištenja na manjim zaslonima i rezolucijama (npr. rezoluciji 1024*768 i manjoj) nije odjednom vidljiva čitava radna ploha, već se za pristup dijelovima sadržaja moraju koristiti horizontalni i vertikalni klizači. Zbog toga bi aplikaciju trebalo oblikovati tako da se veličina elemenata dinamički prilagođava veličini prozora u kojem se aplikacija koristi. To je dosta važno jer se za pristup sadržajima na webu danas sve više koriste mini prijenosna računala za rad na mreži (engl. *netbook*) koja imaju manje zaslone i rezolucije.

Podešavanje ostalih postavki elemenata mape poput boje i debljine linija, boje i izgleda pozadine te izgleda koncepata i veza nije neophodno, ali može pomoći korisnicima u lakšem prihvaćanju aplikacije.

7.3.3.3 Prikaz vezanih resursa

Korisnici u aplikaciji nemaju mjesto na kojem odjednom mogu vidjeti sve resurse vezane uz koncept. Korisnik ima mogućnost pojedinačnog uvida i tada svaku vrstu resursa vidi u svojoj formi. Aplikacija ima mogućnost istovremenog prikazivanja više formi ali je ograničavajući faktor veličina forme. Zbog toga bi trebalo redizajnirati forme, iz njih ukloniti manje važne sadržaje i omogućiti istovremeno prikazivanje više formi u obliku portala.

7.3.3.4 Korištenje HTML sadržaja

Prikazivanje HTML sadržaja unutar aplikacije je vrlo ograničeno, tako da ne postoji mogućnosti uključivanja složenijeg HTML sadržaja. To je posljedica slabosti korištene *Flash* tehnologije. Pošto je mnogo sadržaja koji se koriste u procesu učenja oblikovano kao HTML stranica, oni se prikazuju u novom prozoru web preglednika. Takav način zbunjuje korisnike je moraju prelaziti iz okruženja aplikacije u okruženja klasičnog web preglednika, a web stranica se ovisno o vrsti i verziji web preglednika nekad prikazuje u novom prozoru, a nekad u drugoj kartici istog prozora web preglednika.

Unutar aplikacije HTML sadržaj bilješki uređuje se korištenjem *Flashovog* uređivača teksta koji stvara zastarjeli HTML kod ograničenih mogućnosti (npr. nije moguće umetnuti tablicu ili oblikovati sadržaj pomoću CSS-a). Za izradu HTML sadržaja trebalo bi koristiti uređivač teksta koji stvara standardni HTML 4 kod. Kada bi HTML sadržaji bili izravno čitljivi unutar

aplikacije to bi za korisnika bilo mnogo bolje i time bi se omogućilo umetanje svake vrste HTML sadržaja u okruženju aplikacije, pa tako i vizualnog uređivača teksta.

Djelomično zaobilaženje ograničenja tehnologije moguće je umetanjem HTML *IFrame* objekta iznad transparentnog *Flash* elementa, tako da vizualno izgledaju kao jedan element. U *IFrame* objekt bi se u tom slučaju umetnula HTML stranica s uređivačem teksta koji proizvode ispravan HTML 4 kod. Problem s tim pristupom je što umetanje *IFrame* objekta usporava rad aplikacije, a sama funkcionalnost je dosta nestabilna pa se ne preporučuje za korištenje u praksi [19][25].

Pravi način za rješenje tog problema je izrada problematičnih dijelova aplikacije korištenjem tehnologija koje u potpunosti podržavaju standardni HTML kod (npr. *Ajax*). Na takav način dijelovi aplikacije koji rade s konceptualnom mapom ostali bi realizirani u *Flash* tehnologiji koja je optimalna za takvu vrstu funkcionalnosti dok bi se ostali dijelovi koji se odnose na prijavu korisnika u sustav, administraciju sustava te stvaranja i korištenja obrazovnih materijala realizirali korištenjem nove tehnologije.

7.3.3.5 Povezivanje aplikacije s web preglednikom

Jedna od standardnih funkcionalnosti svakog web preglednika je mogućnost vraćanja korisnika na prethodno posjećenu web stranicu. Obično se to radi pritiskom na tipku za povratak na prethodni sadržaj (engl. *back*). RIA aplikacije standardno ne podržavaju punu funkcionalnost te tipke tako da će se korisnik kada ju pritisne u okruženju aplikacije koja nije posebno podešena, najčešće naći van aplikacije na prethodno posjećenoj web stranici. Najjednostavniji način na koji se mogu spriječiti posljedice slučajnog korištenje te tipke je onemogućavanje njezine funkcionalnosti pomoću JS programskog koda. Time korisnik gubi funkcionalnost vraćanja na prethodnu web stranicu, ali se istovremeno sprječava slučajni izlazak iz aplikacije.

U aplikaciju *Samouk* ugrađena je djelomična podrška korištenju funkcionalnosti *back* tipke koju standardno nudi *Flex* programski okvir (pamti se povijest kretanja po različitim karticama unutar jedne forme i sl.). Rezultat toga je da će korisnik koji slučajno pritisne tu tipku unutar ove aplikacije *ponekad* izaći iz nje. Način na koji je realizirana ta podrška nije zadovoljavajući, pa bi ga u budućnosti trebalo unaprijediti.

7.3.3.6 Podešavanje web servisa

U realiziranoj aplikaciji korisnici iz skupa ponuđenih web servisa biraju one koje žele koristiti. Da bi osobno okruženje bilo u pravom smislu osobno i prilagodljivo, korisnik bi trebao imati mogućnost izbora bilo kojeg web servisa izloženog pomoću svog API-ja. Sadašnji način uključivanja novog web servisa zahtjeva programiranje poslužiteljskog dijela aplikacije. Iako se na takav način funkcionalnost aplikacije može relativno jednostavno proširiti, takav način prilagođavanja sustava nije namijenjen krajnjim korisnicima.

Prilagođavanje korištenju različitih web servisa trebalo bi se realizirati na jednostavniji način. Jedan mogući način je korištenje standardnog upitnog jezika za dohvat podataka. Primjer takvog jezika SQI (*Simple Query Interface*) [95]. Da bi se koristio takav pristup sučelje i

podaci web servisa moraju biti dobro opisani,a davatelj web servisa mora podržavati taj protokol. Broj sustava koji podržavaju SQI za sada je vrlo ograničen, a ograničenje takvog pristupa je što se na takav način podaci mogu samo dohvaćati, a ne i mijenjati.

Drugi način je korištenje posebnog programskog jezika prilagođenog krajnjim korisnicima pomoću kojeg bi oni mogli samostalno birati i podešavati željene web servise. Da bi korisnici mogli koristiti takav jezik on mora biti vrlo jednostavan za korištenje. Primjer takvog jezika je LISL (*Learner Interaction Scripting Language*) koji je izrađen u sklopu projekta razvoja MUPPLE sustava [85]. Kod takvog pristupa korisnici bi mogli podešavati web servise ne samo za dohvat podataka već i za pohranu i brisanje podataka.

Podešavanje web servisa na opisani način namijenjeno je naprednijim korisnicima sustava, a za druge, „obične“ korisnike trebalo bi zadržati postojeći način izbora između ponuđenih servisa i pripremiti skup servisa koji su u potpunosti podešeni. Takav korisnik bi neposredno po prijavi bez ikakvog podešavanja mogao koristiti njihovu funkcionalnost, dok bi napredniji korisnici imali mogućnost daljnjeg podešavanja web servisa po svojim željama.

7.3.3.7 Grupni rad

Trenutna verzija aplikacije *Samouk* pruža korisnicima minimalnu mogućnost grupnog rada po principu „svi ili nitko“. Podaci označeni kao privatni dostupni su samo autoru, a svi ostali podaci su javno dostupni svim prijavljenim korisnicima sustava koji ih mogu koristiti i mijenjati. Jedina funkcionalnost koja je ostala isključivo autoru je mogućnost njihovog brisanja. Tijekom testiranja korisnici su dali primjedbu na slobodno ažuriranje materijala na takav način. Smatraju da je potrebno ograničiti takvo mijenjanje sadržaja ili omogućiti uvid u sve napravljene promjene. Predloženo je da se dozvoli samo komentiranje javno dostupnih materijala, a ne i njihovo mijenjanje. Puni sadržaj nekog materijala u tom slučaju bi se sastojao od izvornog teksta koji je napisao autor i svih komentara drugih korisnika na taj tekst, po principu na kojem npr. funkcionira pisanje bloga.

Drugi način na koji se taj problem može riješiti je verzioniranje sadržaja po principu na koji to rade wiki sustavi. U tom slučaju svaki korisnik može napraviti promjenu u sadržaju, a aplikacija će upamtiti svakog korisnika koji je mijenjao sadržaj i svaku napravljenu promjenu. Uvidom u te podatke moći će se vidjeti koji korisnik je i kada napravio određenu promjenu u sadržaju, a u slučaju potrebe biti će moguće vratiti bilo koju od starih verzija sadržaja.

Funkcionalnost koju bi svakako trebalo uključiti u slijedeću verziju programa je podrška stvaranju grupa. Jedan primjer u kojem je grupni rad poželjan je npr. pisanje članka od strane nekoliko autora. U takvom slučaju nije poželjno da u procesu izrade sudjeluju svi korisnici sustava jer se može postaviti pitanje autorstva i sl. te bi za stvarno korištenje u takvom slučaju trebalo ograničiti broj korisnika koji imaju pristup materijalima.

7.4 Usporedba aplikacije *Samouk* s drugim sustavima za stvaranje PLE-a

U ovom odlomku usporediti će se osobine aplikacije *Samouk* s osobinama četiri sustava za podršku stvaranju PLE-a koji su opisani i međusobno uspoređeni u četvrtom poglavlju.

Rezultati usporedbe prikazani su u tablici 7.1.

Tablica 7.1 Usporedba aplikacije *Samouk* s drugim sustavima za stvaranje PLE-a

	Blog	**Osobni portal**	**mPLE**	**PLEF**	***Samouk***
Tehnologija klijenta	*Ajax*	*Ajax*	*Ajax +Flash*	*Ajax*	*Flash*
Provjera identiteta	*OpenID*, vlastita	Vlastita	Vlastita	*OpenID*	*OpenID*
Višestruki identiteti	Ne	Ne	Ne	Ne	Da
Stvaranje korisničkih sadržaja	Blog	*Widget*	Blog	*Widget*	Interno, pohrana pomoću web servisa
Pridruživanje multimedijalnih materijala	HTML, *widget*	*Widget*	Da	*Widget*	Da
RSS	*Widget*	Da	*Widget*	*Widget*	Da
Komentiranje	Da	Ne	Da	Da	Ne
Korisničke oznake	Da	Ne	Da	Da	Da
Plan učenja	Ne	Ne	Da	Ne	Da
Asinkrona komunikacija	*Widget*	*Widget*	Da	Da	Da
Sinkrona komunikacija	*Widget*	*Widget*	Da	Ne	Da
Uvoz sadržaja	Da	Da	Ne	Ne	Da
Izvoz sadržaja	Da	Da	Ne	Ne	Da
Status	Funkcionalno	Funkcionalno	U fazi izrade	U fazi izrade	U fazi izrade

Aplikacija *Samouk* izrađena je u *Flash* tehnologiji što nije čest slučaj kod drugih, sličnih aplikacija koje se koriste u praksi. Posljedica korištenja tog pristupa je nemogućnost indeksiranja sadržaja od strane web pretraživača i nemogućnost učinkovitog prikaza HTML sadržaja u okruženju aplikacije. Identitet korisnika utvrđuje se pomoću *OpenIDja* korisnika kao i u dijelu ostalih aplikacija, a rješenje koje ne podržava niti jedan od navedenih sustava je

mogućnost pridruživanja više različitih *OpenID* oznaka jednom korisniku. Na takav način se osigurava korištenje sustava ako neki od davatelja identiteta nije u funkciji.

Tekstualni sadržaji se pohranjuju u okruženje izabranog web servisa. Pritom korisnik može izabrati različite servise (mikroblog, uredsku aplikaciju ili servis za pohranu bilješki). Druga okruženja su ograničena na pohranu podataka na jedan način, a podatke pohranjuju unutar vlastitog okruženja. Pri stvaranju tekstualnih sadržaja kao i u drugim okruženjima koristi se vizualni uređivač teksta. Multimedijalni materijali (slike, video, zvuk i sl.) vežu se pomoću URI-ja uz određeni sadržaj na sličan način kao kod ostalih okruženja. Prednost pri pronalaženju tih materijala u okruženju realizirane aplikacije je u tome što korisnik podatke može pretraživati unutar okruženja istovremeno koristeći više izabranih pretraživača. Dohvaćanje RSS sadržaja podržano je na sličan način kao i kod ostalih sustava.

Komentiranje sadržaja nije podržano u trenutnoj verziji aplikacije, dok je označavanje sadržaja korisničkim oznakama djelomično podržano kod označavanja materijala pridruženih konceptima.

Aplikacija *Samouk* omogućuje izradu plana učenje u grafičkom obliku na sličan način kao što to omogućuje mPLE aplikacija. Razlika je što mPLE ima bogatije sučelje i omogućuje vezanje materijala uz čvorove mape, dok je u ovoj aplikaciji potrebno pristupiti formi u kojoj su u tabličnom obliku prikazani vezani materijali. Izravno pridruživanje je za korisnika vizualno ljepše i jednostavnije za korištenje. Problemi se javljaju kada je određenom čvoru potrebno priložiti više različitih materijala i u tom slučaju stvorena mapa vrlo brzo postaje nepregledna. Kada se svi vezani materijali prikazuju u posebnoj formi, čvoru se može pridružiti neograničeno mnogo različitih materijala, a zbog toga sam grafički prikaz neće postati složeniji.

Asinkrona i sinkrona komunikacija između korisnika podržana je na sličan način kao u drugim okruženjima. Razlika je što kod sinkrone komunikacije čavrljanjem korisnik može jednostavno određene dijelove komunikacije pohraniti u obliku bilješke.

Aplikacija podržava ograničen izvoz i uvoz podataka konceptualne mape u CXL formatu, ali ne podržava razmjenu kompletne strukture podataka (npr. sigurnosnu pohranu mape i svih podataka u obliku XML datoteke) poput bloga ili osobnog portala.

Aplikacija *Samouk* u globalu podržava većinu funkcionalnosti kao i druge s kojima se uspoređuje. Neke od uočenih prednosti su mogućnost vezanja više različitih *OpenID* identiteta uz korisnika, mogućnost pohrane podataka koristeći web servise po izboru korisnika, vizualno predstavljanje strukture znanja u obliku konceptualne mape te pridruživanje materijala čvorovima mape. Način funkcioniranja, jednostavnost korištenja i vizualni dojam aplikacije zaostaju za rješenjima koja se koriste u praksi i na tom području bi trebalo napraviti dorade u skladu s primjedbama koje se dobiju od korisnika tijekom daljnjeg testiranja.

7.5 Daljnji razvoj

Pri inicijalnom testiranju aplikacije *Samouk* nisu uočeni veći problemi u funkcioniranju, a otkriven je i prijavljen određen broj grešaka. Nakon što se prijavljene greške isprave, aplikacija će biti pogodna za stvaranje PLE-a koje će se moći koristiti pri samostalnom učenju.

U slijedećoj fazi razvoja u skladu s primjedbama uočenim tijekom testiranja napraviti će se izmjene funkcionalnih zahtjeva opisanih u petom poglavlju. Nakon toga se planira dorada aplikacije u skladu s novim zahtjevima i njezino puštanje u testni rad na većem broju korisnika. Planira se testiranje na dvije skupine korisnika. Jedna skupina obuhvaćati će nastavnike i studente preddiplomskih i diplomskih studija Sveučilišta u Dubrovniku, a druga će obuhvaćati korisnike Interneta koji se dobrovoljno jave za testiranje aplikacije. Kako bi se u testiranje aplikacije omogućilo uključivanje i korisnika van RH, planira se u aplikaciji napraviti dvojezično korisničko sučelje (hrvatsko i englesko). Testiranje aplikacije na većem broju korisnika dati će informaciju o tome u kojoj mjeri je aplikacija po svojim osobinama spremna za korištenje u stvarnom radu, a povratne informacije koje se dobiju od korisnika tijekom tog testiranja iskoristiti će se za dorade i daljnji razvoj modela, a i same aplikacije.

U daljnjem razvoju aplikacije *Samouk* poseban naglasak staviti će se na tri područja:

- podršku grupnom radu - Podrška grupnom radu u aplikaciji trenutno je minimalna, a rad u takvom okruženju jedna je od osnovnih osobina koje pomaže pri samostalnom učenju.
- jednostavnije podešavanje korištenih web servisa - Da bi okruženje bilo u pravom smislu riječi osobno, korisnik koji to želi treba imati mogućnost jednostavnog podešavanja većine postavki okruženja. Okruženje za dohvat i pohranu materijala koristi web servise, a danas se na webu svakodnevno pojavljuju novi servisi s funkcionalnostima koje se mogu iskoristiti u procesu samostalnog učenja. Zbog toga je vrlo važno korisniku omogućiti izbor između različitih web servisa i njihovo jednostavno podešavanje.
- poboljšanje korištenja HTML sadržaja u okruženju aplikacije - Većina sadržaja u elektroničkom obliku koji se nalaze na webu dolazi u obliku HTML stranice. Okruženje koje se bazira na prikupljanju takvih materijala *mora* imati mogućnost dobrog prikazivanja HTML sadržaja. Time će se omogućiti i jednostavnije korištenje funkcionalnosti koje se planiraju uključiti u nove verzije HTML-a (npr. crtanje dvodimenzionalne grafike ili prikazivanje multimedijalnih sadržaja bez potrebe za korištenjem posebnih vanjskih dodataka [103]).

8 ZAKLJUČAK

U posljednjih desetak godina dominantan način stjecanja znanja postalo je cjeloživotno samousmjereno učenje u kojem učenici samostalno određuju što će, kada i na koji način učiti. U tom procesu mogu koristiti mnoštvo materijala i aplikacija dostupnih na webu. Primjena korištenih resursa je puno učinkovitija ako se koriste unutar specifičnog računalom podržanog okoliša za učenje.

U ovom radu opisana je realizirana programska potpora stvaranju osobnog okoliša za učenje. U teorijskom dijelu napravljen je pregled tehnologija i sustava koji su utjecali na ideju oblikovanja osobnih okoliša za učenje. Naglasak je stavljen na elemente koji su korišteni prilikom izrade modela i aplikacije *Samouk*, opisanih u praktičnom dijelu rada. Izrađena aplikacija testirana je u praktičnom radu, a rezultati testiranja iskoristiti će se za poboljšanje modela i aplikacije.

Aplikacija *Samouk* omogućuje stvaranje osobnog okoliša za učenje koji učenici mogu koristiti u procesu stvaranja novog znanja. Na početku tog procesa učenici izrađuju plan koji je vizualno predstavljen u obliku konceptualne mape. Praktično korištenje mapa tijekom učenja pokazalo je da prikaz podataka u tom obliku učenicima pomaže pri shvaćanju strukture i odnosa između njezinih dijelova. Učenje je iterativan proces tijekom kojeg se plan nadograđuje, mijenja i prilagođava novim spoznajama do kojih učenici dolaze učenjem iz prikupljenih materijala i komunikacijom s drugim učenicima. Materijali se dobavljaju i pohranjuju na webu korištenjem web servisa. Tijekom tog procesa plan učenja s pridruženim materijalima postepeno se pretvara u mapu znanja i postaje repozitorij u kojem su trajno pohranjene informacije o određenom području.

Aplikacija *Samouk* sastoji se od klijentskog i poslužiteljskog dijela. Klijentski dio izrađen je u obliku *Flash* RIA-e kako bi se korisniku pojednostavnio rad i omogućila veća funkcionalnost slična stolnim aplikacijama. Poslužiteljski dio za dohvat i pohranu podataka koristi različite Web 2.0 servise po izboru korisnika.

Zadani cilj ovog magistarskog rada, izrada konceptualnog modela i realizacija aplikacije za potporu stvaranju osobnog okoliša za učenje je ostvaren. Tijekom provedenog testiranja aplikacije izrađeni model se pokazao dobro strukturiranim, a aplikacija korisnim alatom koji može pomoći pri stvaranju osobnog okoliša za učenje. Tijekom testiranja uočeni su određeni nedostaci modela i aplikacije koji će biti ispravljeni u budućim verzijama.

Nakon dorade uočenih nedostataka planira se provesti testiranje aplikacije na većem broju korisnika koje bi trebalo dati informaciju o tome u kojoj mjeri je aplikacija spremna za korištenje u stvarnom radu.

9 LITERATURA

[1] R. Adler, J. Stipins, M. Ohye, *Improved Flash indexing*, Official Google Webmaster Central Blog, http://googlewebmastercentral.blogspot.com/2008/06/improved-flash-indexing.html (22.8.2009.)

[2] M. Ally, *Foundations of Educational Theory for Online Learning*, in T. Anderson, F. Elloumi, editors, The Theory and Practice of Online Learning, 2nd ed, AU Press, Athabasca University, 2008.

[3] S. Ash, *MoSCoW Prioritisation Briefing Paper*, DSDM Consortium, http://www.dsdm.org/knowledgebase/details/165/moscow-prioritisation-briefing-paper.html (20.08.2008.)

[4] G. Attwell, J. Bimrose, A. Brown et al., *Maturing Learning: Mashup Personal Learning Environments*, Proceedings of the First International Workshop on Mashup Personal Learning Environments (MUPPLE08), Maastricht, Netherlands, September 17, 2008., http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-388/attwell.pdf (2.8.2009.)

[5] A-L. Barabasi, *U mreži*, Naklada Jesenski i Turk, Zagreb, 2006.

[6] T. Berners-Lee, *Information Management: A Proposal*, CERN March 1989, May 1990. http://www.w3.org/History/1989/proposal.rtf (10.8.2009.)

[7] T. Berners-Lee, J. Hendler, O. Lassila, *The Semantic Web*, Scientific American Magazine, May 2001., http://www.scientificamerican.com/article.cfm?id=the-semantic-web (11.9.2009.)

[8] I. Blees, M. Rittberger, *Web 2.0 Learning Environment: Concept, Implementation, Evaluation*, eLearning Papers, No. 15, 2009., http://www.elearningpapers.eu/ (10.7.2009.)

[9] P. Brusilovsky, *Methods and techniques of adaptive hypermedia,* User Modeling and User Adapted Interaction, Vol. 6, No. 2-3, 1996., pp 87-129. http://www2.sis.pitt.edu/~peterb/papers/UMUAI96.pdf (12.5.2009.)

[10] V. Bush, *As We May Think*, The Atlantic Mothly, July 1945., http://www.theatlantic.com/doc/194507/bush (10.9.2009.)

[11] A.J. Cañas, G. Hill, L, Bunch, et al., *KEA: A knowledge exchange architecture based on web services, concept maps and cmaptools*, Proceedings of the Second International Conference on Concept Mapping (CMC 2006), San José, Costa Rica, 2006., http://cmc.ihmc.us/cmc2006Papers/cmc2006-p234.pdf (2.8.2009.)

[12] M.J. Carnot, B. Dunn, A. J. Cañas, et al., *Concept Maps vs. Web Pages for Information Searching and Browsing*, http://www.ihmc.us/users/acanas/Publications/CMapsVSWebPagesExp1/CMapsVSWebPagesExp1.htm (5.8.2009.)

[13] M. A. Chatti, M. Jarke, M. Specht, *PLEF: A Conceptual Framework for Mashup Personal Learning Environments*, IEEE Learning Technology Newsletter, Vol. 11, No. 3, July 2009., http://www.ieeetclt.org/issues/july2009/index.html (31.8.2009.)

[14] M. A. Chatti, *Personal Environments Loosely Joined*, http://mohamedaminechatti.blogspot.com/2007/01/personal-environments-loosely-joined.html (12.8.2009.)

[15] M. A. Chatti, M. Jarke, Z. Wanget et al., *SMashup Personal Learning Environments*, 2nd Workshop on Mash-UP Personal Learning Environments (MUPPLE-09), Nice, France, September 29 - October 2, 2009., http://www-i5.informatik.rwth-achen.de/lehrstuhl/staff/chatti/download/Mupple09-CJWS.pdf (16.09.2009.)

[16] I. Chen, *Behaviorism and Developments in Instructional Design and Technology*, in P. Rogers, G. A. Berg, J. V. Boettecher et al., editors, Encyclopedia of Distance Learning, 2nd ed., IGI global, 2009.

[17] J. Cross, *Informal Learning: Rediscovering the Natural Pathways That Inspire Innovation and Performance*, John Wiley & Sons, 2007.

[18] J. Cross, *Stuff that works*, http://www.informl.com/2006/11/19/stuff-that-works/ (9.9.2009.)

[19] B. Deitte, *Don't Use IFrames for HTML in Flex*, http://www.deitte.com/archives/2008/07/dont_use_iframe.htm (1.4.2009.)

[20] K. Dobbs, *Simple Moments of Learning*, Training Vol. 37, No. 1, 2000., pp 52-58.

[21] S. Downes, http://www.downes.ca/ (07.09.2009.)

[22] S. Downes, *Authentication and Identification*, International Journal of Instructional Technology and Distance Learning, Vol. 2, No. 10, 2005., http://www.itdl.org/Journal/Oct_05/article01.htm (15.12.2008.)

[23] S. Downes, *E-learning 2.0*, eLearn magazine, http://www.elearnmag.org/subpage.cfm?section=articles&article=29-1 (12.3.2009.)

[24] S. Dziadosz, R. Chandrasekar, *Do Thumbnail Previews Help Users Make Better Relevance Decisions about Web Search Results?*, Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, Tampere, Finland, August 2002., pp 365–366.

[25] J. Everett-Church, *Wmode Woes*, http://justin.everett-church.com/index.php/2006/02/23/wmode-woes/ (1.4.2009.)

[26] C. Fallon, S. Brown, *E-learning Standards: a Guide to Purchasing, Developing and Deploying Standards-Conformant E-learning*, CRC Press, 2003.

[27] J. Farmer, A. Bartlett-Bragg, *Blogs @ anywhere: High fidelity online communication*, ASCILITE 2005., Brisbane, Australia, http://www.ascilite.org.au/conferences/brisbane05/blogs/proceedings/22_Farmer.pdf (12.6.2009.)

[28] J. Feiler, *How to Do Everything with Web 2.0 Mashups*, McGraw-Hill Osborne Media, 2008.

[29] K. Fenning, *Cohort Based Learning: Application to Learning Organizations and Student Academic Success*, College Quarterly, Vol. 7, No. 1, 2004., http://www.senecac.on.ca/quarterly/2004-vol07-num01-winter/fenning.html (12.5.2009.)

[30] R. T. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, Doctoral dissertation, University of California, Irvine, 2000., http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf (12.5.2009.)

[31] R. T. Fielding, J. Gettys, J. Mogul et al., *Hypertext Transfer Protocol - HTTP/1.1*, Internet Official Protocol Standards, June 1999., http://www.w3.org/Protocols/rfc2616/rfc2616.html (4.9.2009.)

[32] J. J. Garrett, *Ajax: A New Approach to Web Applications*, http://www.adaptivepath.com/ideas/essays/archives/000385.php (12.2.2009.)

[33] J. Giles, *Internet Encyclopaedias Go Head to Head*, Nature, Vol. 438, No. 7070, December 15 2005., pp 900-901, http://www.nature.com/nature/journal/v438/n7070/full/438900a.html (1.8.2009.)

[34] J. Grossman, R. Hansen, P. D. Petkov et al., *Cross Site Scripting Attacks: XSS Exploits and Defense*, Syngress Publishing, 2007.

[35] M. Harrison, *13 Ways of Managing Informal Learning*, Kineo Insight, Kineo, 2005. http://www.kineo.co.uk/publications/insight-reports-home-page.html (27.12.2007.)

[36] N. Hoić-Božić, V. Mornar, *AHyCo: a Web-Based Adaptive Hypermedia Courseware System*, Journal of Computing and Information Technology – CIT, Vol. 13, No. 3, 2005., pp 165–176.

[37] K. Hoyt, *Introducing Adobe AIR for Ajax developers*, Adobe developer connection, http://www.adobe.com/devnet/air/ajax/articles/air_ajax_developers.html (16.8.2009.)

[38] M. V. Hejmadi, *Improving the Effectiveness and Efficiency of Teaching Large Classes: Development and Evaluation of a Novel e-Resource in Cancer Biology*, Bioscience Education Vol. 9, June 2007., http://www.bioscience.heacademy.ac.uk/journal/vol9/beej-9-2.aspx (18.7.2009.)

[39] W. Huitt, *Constructivism*, Educational Psychology Interactive, Valdosta State University, Valdosta, GA, http://teach.valdosta.edu/whuitt/col/cogsys/construct.html (1.8.2009.)

[40] A. Jafari, P. McGee, C. Carmean, *Managing Courses, Defining Learning: What Faculty, Students, and Administrators Want*, EDUCAUSE Review, Vol. 41, No. 4, July/August 2006, pp 50–71.

[41] C. Kadushin, *Introduction to Social Network Theory*, Draft of Chapter 2. Some Basic Network Concepts and Propositions, http://home.earthlink.net/~ckadushin/Texts/Basic%20Network%20Concepts.pdf (11.8.2009.)

[42] D. Kalpić, J. Anzil, H. Zoković, *From the Traditional to a Digital Academic Library*, CoLIS3, Zagreb, Benja, 1999., pp 317-321.

[43] A. Keen, *The Cult of the Amateur: How Today's Internet is Killing Our Culture*, Doubleday, 2007.

[44] B. Kennedy, C. Musciano, *HTML & XHTML: The Definitive Guide*, 5th Edition, O'Reilly, 2002.

[45] R. S. Kennet, E. R. Baker, *Software Process Quality*, Marcel Dekker Inc., New York – Basel, 1999.

[46] J. Kolbitsch, H. Mauer, *Community Building around Encyclopaedic Knowledge*, Journal of Computing and Information Technology - CIT, Vol. 14, No. 3, 2006., pp 175-190.

[47] Y. Lee, A. L. Baylor, D. W. Nelson, *Supporting Problem-Solving Performance Through the Construction of Knowledge Maps*, Journal of Interactive Learning Research, Vol. 16 No. 2, 2005., pp 117–131.

[48] J. Leskovec, E. Horvitz, *Planetary-Scale Views on an Instant-Messaging Network*, Microsoft Research Technical Report, June 2007, http://arxiv.org/pdf/0803.0939v1 (6.8.2009.)

[49] O. Liber, *Colloquia - a conversation manager*, Campus-wide Information System (CWIS), Vol. 17, No. 2, 2000., pp 56-62.

[50] R. Lubensky, *The present and future of Personal Learning Environments (PLE)*, http://www.deliberations.com.au/2006/12/ present-and-future-of-personal-learning.html (12.3.2009.)

[51] E. Maler, D. Reed, *The Venn of Identity: Options and Issues in Federated Identity Management*, IEEE Security and Privacy, Vol. 6, No. 2, 2008., pp 16-23.

[52] M. Martin, *My Personal Learning Environment*, http://michelemartin.typepad.com/ thebambooprojectblog/2007/04/my_personal_lea.html (12.7.2009.)

[53] H. Mauer, K. Tochtermann, *On a New Powerful Model for Knowledge Management and its Applications*, Journal of Universal Computer Science, Vol. 8, No. 1, 2002., http://www.jucs.org/jucs_8_1/on_a_new_powerful/Maurer_H.html (20.8.2009.)

[54] A. McAfee*, Enterprise 2.0: The Dawn of Emergent Collaboration*, MIT Sloan Management review. Vol. 47, No. 3, 2006., pp 21-28. http://adamkcarson.files.wordpress.com/2006/12/enterprise_20_-_the_dawn_of_emergent_collaboration_by_andrew_mcafee.pdf (1.8.2009.)

[55] M. Milicevic, K. Zubrinic, I. Zakarija, *Dynamic Approach to the Construction of Progress Indicator for a Long Running SQL Queries*, International Journal of Computers, Vol. 2, No 4, 2008, http://www.naun.org/journals/computers/ijcomputers-108.pdf, (2.6.2009.)

[56] C. D. Milligan, P. Beauvoir, M. W. Johnson et al., *Developing a Reference Model to Describe the Personal Learning Environment*, in W. Nejdl, K. Tochtermann, editors, Innovative Approaches for Learning and Knowledge Sharing, Lecture Notes in Computer Science Vol. 4227, Springer, 2006., pp 506-511.

[57] F. Moritz, *Rich Internet Applications (RIA): A Convergence of User Interface Paradigms of Web and Desktop Exemplified by JavaFX*, Diploma Thesis, University of Applied Science Kaiserslautern Fachhochschule, Kaiserslautern, Germany, January 2008., http://www.flomedia.de/diploma/documents/DiplomaThesisFlorianMoritz.pdf (2.4.2009.)

[58] A. Nanda, *Identity Selector Interoperability Profile V1.0*, normative specification, Microsoft, 2007., http://download.microsoft.com/download/1/1/a/11ac6505-e4c0-4e05-987c-6f1d31855cd2/Identity-Selector-Interop-Profile-v1.pdf (12.8.2009.)

[59] T. H. Nelson, *Complex information processing: a file structure for the complex, the changing and the indeterminate*, Proceedings of the 20th ACM national conference, Cleveland, Ohio, United States, 1965., pp 84–100.

[60] J. Nielsen, *Usability Engineering*, Morgan Kaufmann, San Francisco,1994.

[61] J. D. Novak, A.J.Cañas, *The Theory Underlying Concept Maps and How to Construct and Use Them*, Technical Report, http://cmap.ihmc.us/Publications/ResearchPapers/TheoryCmaps/TheoryUnderlyingConceptMaps.htm (12.6.2009.)

[62] W. O'Donohue, R. Kitchener, editors, *Handbook of Behaviorism*, Academic Press, 1997.

[63] T. O'Reilly, *What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software*, http://oreilly.com/web2/archive/what-is-web-20.html (10.12.2008.)

[64] N.Pastuović, *Edukologija*, Znamen, Zagreb, 1999.

[65] J. Quarter, H. Midha, *Informal Learning Processes in a Worker Co-operative*, NALL Working Paper No. 37, Ontario Institute for Studies in Education of the University of Toronto, Toronto, 2001., http://www.oise.utoronto.ca/depts/sese/csew/nall/res/37workerscoop.pdf (12.6.2009.)

[66] L. Richardson, S. Ruby, *RESTful Web Services*, O'Reilly, 2007.

[67] M. Rosić, V. Glavinić, S. Stankov, *Intelligent Tutoring Systems for the New Learning Infrastructure*, in M. D. Lytras and A. Naeve, editors, Intelligent Learning Infrastructure for Knowledge Intensive Organization: A Semantic Web Perspective, August 2005, pp 225-250.

[68] W. B. Sanders, C. Cumaranatunge, *ActionScript 3.0 Design Patterns*, O'Reilly, 2007.

[69] C. Severance, J. Hardin, A. Whyte, *The coming functionality mash-up in Personal Learning Environments*, Interactive Learning Environments, Vol. 16, No. 1, April 2008., pp 47-62.

[70] L. Shepard, *How to accept OpenID in a popup without leaving the page*, http://www.sociallipstick.com/2009/02/04/how-to-accept-openid-in-a-popup-without-leaving-the-page/ (5.6.2009.)

[71] A. P. Sheth, K. Gomadam, J. Lathem, *SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups*, IEEE Internet Computing, Vol. 11 No. 6, 2007., pp 91-94.

[72] G. Siemens, Connectivism: *A Learning Theory for the Digital Age*, International Journal of Instructional Technology and Distance Learning, Vol. 2, No. 1, 2005., http://www.itdl.org/Journal/Jan_05/article01.htm (2.4.2009.)

[73] B. F. Skinner, *Teaching Machines*, Science, Vol. 128, No. 3330, October 1958., pp 969-977, http://www.bfskinner.org/teachingmachines1958.pdf (28.8.2009.)

[74] C. Smart, S. Holyfield, *Building the ELF (e-Learning Framework) Community*, A report on the JISC/CETIS Conference on e-Learning Tools, Standards and Systems , Oxford, 4/5 November 2004., http://www.elearning.ac.uk/news_folder/features/oxford_conf/document_view (12.8.2009.)

[75] J. Snell, *chmod 777 web*, https://www.ibm.com/developerworks/mydeveloperworks/blogs/jasnell/entry/chmod_777_web (21.8.2009.)

[76] S. Souder, *Even Faster Web Sites*, O'Reilly, 2009.

[77] C. Talbot, *Studying at a distance: A guide for students*, 2nd ed., Open University Press, McGraw-Hill, 2007.

[78] L. Teemu, A. Raami, S. Mielonen et al., *FLE - Tools Prototype: A WWW-based Learning Environment for Collaborative Knowledge Building*, ENABLE99 Enabling Network-Based Learning, International Conference, Helsinki, 1999., http://mlab.taik.fi/fle/research/enable.html (20.8.2009.)

[79] M. van Harmelen, *Design Trajectories: Four Experiments in PLE Implementation*, Interactive Learning Environments, Vol. 16, No. 1, April 2008., pp 35–46.

[80] M. van Harmelen, *Manchester PLE demos during ALT-C 2009*, http://hedtek.com/?p=164 (13.9.2009.)

[81] M. van Harmelen, *Personal Learning Environments*, http://octette.cs.man.ac.uk/jitt/index.php/Personal_Learning_Environments (12.7.2009.)

[82] M. van Harmelen, M. Metcalfe, D. Randall, *The Manchester PLE Project*, JISC Emerge Benefits Realisation Report, http://reports.jiscemerge.org.uk/View-document-details/5-The-Manchester-PLE-Project.html (5.9.2009.)

[83] T. Vander Wal, *Folksonomy*, http://vanderwal.net/folksonomy.html (10.8.2009.)

[84] M. Weller, *Virtual Learning Environments: Using, choosing and developing your VLE*, Routlege, New York, 2007.

[85] F. Wild, F.Mödritscher, S. Sigurdarson, *Mash-Up Personal Learning Environments*, Final Report, January 2009., http://www.icamp.eu/wp-content/uploads/2009/01/d34_icamp_final.pdf (5.3.2009.)

[86] S. Wilson, P. Sharples, D. Griffiths, *Distributing education services to personal and institutional systems using Widgets*, Proceedings of the First International Workshop on Mashup Personal Learning Environments (MUPPLE08), Maastricht, The Netherlands, September 17, 2008., http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-388/wilson.pdf (2.8.2009.)

[87] S. Wilson, *Future VLE - The Visual Version*, http://zope.cetis.ac.uk/members/scott/blogview?entry=20050125170206 (3.3.2009.)

[88] S. Wilson, O. Liber, M. Johnson et al., *Personal Learning Environments: Challenging the dominant design of educational systems*, ECTEL Conference, Crete, 2006, http://hdl.handle.net/1820/727 (02.08.2009.)

[89] D. Winer, *XML-RPC Specification*, last update 30.6.2003., http://www.xmlrpc.com/spec (27.8.2009.)

[90] A.Woodruff, R. Rosenholtz, J.B. Morrison, *A comparison of the use of text summaries, plain thumbnails, and enhanced thumbnails for Web search tasks*, Journal of the American Society for Information Science and Technology, Vol. 53, No. 2, 2002, pp 172-185

[91] R. Yee, Pro *Web 2.0 Mashups: Remixing Data and Web Services*, Apress, 2008.

[92] K. Žubrinić, D. Kalpić, *The Web as Personal Learning Environment,* Proceedings of Conference MIPRO CE 2008., May 26 – 30, 2008, Opatija, Croatia, pp 36-41

[93] J. Žufić, Janko, D. Kalpić, *More Efficient Learning on Web Courseware Systems?*, Proceedings of E-Learn - World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education, Qubec City, Canada, 15-19.10.2007., pp 6707-6716.

[94] ***, *A Memorandum on Lifelong Learning*, SEC 1832, Commission of the European Communities, Brussels, 2000., http://web.aoo.hr/Documents/ 9)%20A%20memorandum%20on%20life%20long%20learning.pdf (4.3.2009.)

[95] ***, A *Simple Query Interface Specification for Learning Repository*, Workshop Agreement, European Committee for Standardization, November, 2005., ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/WS-LT/CWA15454-00-2005-Nov.pdf (2.9.2009.)

[96] ***, *Action Message Format -- AMF 3*, Adobe Systems Inc., http://download.macromedia.com/pub/labs/amf/amf3_spec_121207.pdf (2.3.2009.)

[97] ***, *ADL Guidelines for Creating Reusable Content with SCORM 2004*, http://www.adlnet.gov/Technologies/scorm/SCORMSDocuments/ SCORM%20Resources/ADLGuidelines_V1PublicComment.zip (1.9.2009.)

[98] ***, *An XML-based Language for Describing the Content of Cmaps*, IHMC,http://cmap.ihmc.us/xml/CXL.html (7.4.2009.)

[99] ***, *Cross-domain policy file specification*, Adobe Systems Inc., http://www.adobe.com/devnet/articles/crossdomain_policy_file_spec.html (2.6.2009.)

[100] ***, *Draft Standard for Learning Object Metadata*, IEEE, 15 July 2002, http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf (1.9.2009.)

[101] ***, *Fatally Flawed Refuting the Recent Study on Encyclopedic Accuracy by the Journal Nature*, Encyclopaedia Britannica Inc., March 2006., http://corporate.britannica.com/britannica_nature_response.pdf (1.8.2009.)

[102] ***, *Flash Player Penetration survey*, Millward Brown Inc., June 2009., http://www.adobe.com/products/player_census/flashplayer/ (12.7.2009.)

[103] ***, *HTML 5 A vocabulary and associated APIs for HTML and XHTML*, W3C Working Draft, 25 August 2009., http://www.w3.org/TR/html5/ (20.9.2009.)

[104] ***, *IMS Global Learning Consortium Specifications*, http://www.imsglobal.org/specifications.html (9.9.2009.)

[105] ***, *International Social Media Research, Wave 3*, March 2008., Universal Mccann, http://www.slideshare.net/mickstravellin/ universal-mccann-international-social-media-research-wave-3 (20.8.2009.)

[106] ***, *Internet World Stats: Usage and Population Statistics*, Miniwatts Marketing Group, http://www.internetworldstats.com/ (31.7.2009.)

[107] ***, *Introducing JSON*, http://www.json.org/ (13.8.2009.)

[108] ***, *JanRain PHP OpenID library*, http://openidenabled.com/ (7.7.2009.)

[109] ***, *OpenID*, http://openid.net/ (2.7.2009.)

[110] ***, *Relying Party Stats as of July 1, 2009*., JanRain Blog, http://blog.janrain.com/ 2009/07/relying-party-stats-as-of-july-1-2009.html (1.9.2009.)

[111] ***, *RIA statistics*, DreamingWell, July 2009., http://riastats.com/ (12.7.2009.)
[112] ***, *SAML V2.0 Executive Overview*, Draft 01, OASIS, http://www.oasis-open.org/committees/download.php/13525/sstc-saml-exec-overview-2.0-cd-01-2col.pdf (5.9.2009.)
[113] ***, *Service-Oriented Architecture Ontology*, Draft Technical Standard v.2.0, The Open Group, 2008., http://www.opengroup.org/projects/soa-ontology/doc.tpl?CALLER=index.tpl&gdid=16940 (22.8.2009.)
[114] ***, *Statističke informacije 2009*., Državni zavod za statistiku Republike Hrvatske, Zagreb, 2009., http://www.dzs.hr/Hrv_Eng/StatInfo/pdf/StatInfo2009.pdf (31.7.2009.)
[115] ***, *Strategija i akcijski plan obrazovanja odraslih*, Vlada Republike Hrvatske, Povjerenstvo za obrazovanje odraslih, Zagreb, Studeni 2004. http://www.strategija.hr/lgs.axd?t=16&id=203 (12.5.2009.)
[116] ***, *The size of the world wide web*, WorldWideWebSize, http://www.worldwidewebsize.com/ (28.8.2009.)
[117] ***, *Widgets 1.0: Packaging and Configuration*, W3C Candidate Recommendation, 23 July 2009., http://www.w3.org/TR/widgets/ (21.9.2009.)
[118] ***, *Wikipedia: Blog*, http://en.wikipedia.org/wiki/Blog (10.8.2009.)
[119] ***, *Wikipedia: History of Personal Learning Environment*, http://en.wikipedia.org/wiki/History_of_personal_learning_environments (2.9.2009.)
[120] ***, *Wikipedia: History of Virtual Learning Environments*, http://en.wikipedia.org/wiki/History_of_virtual_learning_environments (1.9.2009.)
[121] ***, *Wikipedia: Plato (computer system)*, http://en.wikipedia.org/wiki/PLATO_System (1.9.2009.)
[122] ***, *Wikipedia: Social bookmarking*, http://en.wikipedia.org/wiki/Social_bookmarking (22.8.2009.)
[123] ***, *Wikipedia: Wiki*, http://en.wikipedia.org/wiki/Wiki (10.8.2009.)
[124] ***, *XMLHttpRequest*, W3C Working Draft 20, W3C, August 2009, http://www.w3.org/TR/XMLHttpRequest/ (2.9.2009.)
[125] ***, *YUI Library: Connection Manager*, http://developer.yahoo.com/yui/connection/ (18.7.2009.)
[126] ***, *Zend Framework*, Zend Technologies Ltd., http://framework.zend.com/ (7.9.2009)

10 POPIS INTERNETSKIH ADRESA

[URI1] *43Things*, http://www.43things.com/ (12.9.2009.)
[URI2] *Amazon*, http://www.amazon.com/ (2.7.2009.)
[URI3] *Amazon Simple Storage Service*, http://aws.amazon.com/s3/ (7.9.2009.)
[URI4] *AOL*, http://www.aol.com/ (12.8.2009.)
[URI5] *BlipTV*, http://www.bliptv.com/ (12.7.2009.)
[URI6] *Blogger*, www.blogger.com/ (15.9.2009.)
[URI7] *Box*, http://www.box.net/ (12.9.2009.)
[URI8] *Delicious*, http://www.delicious.com/ (20.9.2009.)
[URI9] *Docstoc*, http://www.docstoc.com/ (7.8.2009.)
[URI10] *Elgg*, http://www.elgg.com/ (12.9.2009.)
[URI11] *Facebook*, http://www.facebook.com/ (2.8.2009.)
[URI12] *Flickr*, http://www.flickr.com/ (3.9.2009.)
[URI13] *Google*, http://www.google.com (20.9.2009.)
[URI14] *Google Alerts*, http://www.google.com/alerts/ (18.9.2009.)
[URI15] *Google Docs*, http://docs.google.com/ (20.9.2009.)
[URI16] *Google Reader*, http://www.google.com/reader/ (21.9.2009.)
[URI17] *Google Spreadsheets*, http://docs.google.com/?pli=1#spreadsheets/ (21.9.2009.)
[URI18] *iGoogle*, http://www.google.com/ig (14.8.2009.)
[URI19] *JavaFX*, http://javafx.com/ (12.6.2009.)
[URI20] *LinkedIn*, http://www.linkedin.com/ (2.9.2009.)
[URI21] *Microsoft Silverlight,* http://silverlight.net/ (12.6.2009.)
[URI22] *Moodle*, http://www.moodle.org/ (4.8.2009.)
[URI23] *MUPPLE*, http://mupple.org/ (17.9.2009.)
[URI24] *MyOpenID*, http://www.myopenid.com/ (12.8.2009.)
[URI25] *MySpace*, http://www.myspace.com/ (2.8.2009.)
[URI26] *Netvibes*, http://www.netvibes.com/ (14.8.2009.)
[URI27] *Pageflakes*, http://www.pageflakes.com/ (14.8.2009.)
[URI28] *Picasa*, http://picasa.google.com/ (1.9.2009.)
[URI29] *PLEF*, http://eiche.informatik.rwth-aachen.de:3333/PLEF/index.jsp (2.9.2009.)
[URI30] *Sakai*, http://sakaiproject.org/portal (20.9.2009.)
[URI31] *Scribd*, http://www.scribd.com/ (21.7.2009.)
[URI32] *Simpy*, http://www.simpy.com/ (6.8.2009.)
[URI33] *Skype*, http://www.skype.com/ (1.8.2009.)
[URI34] *SlideShare*, http://www.slideshare.com/ (7.9.2009.)
[URI35] *Tumblr*, http://www.tumblr.com/ (11.9.2009.)
[URI36] *Twitter*, http://www.twitter.com/ (13.7.2009.)
[URI37] *Verisign,* http://www.verisign.com/ (12.8.2009.)
[URI38] *Wikipedia*, http://www.wikipedia.org/ (21.9.2009.)
[URI39] *Wikispace*, http://www.wikispaces.com/ (15.9.2009.)
[URI40] *Wordpress*, http://wordpress.com/ (15.9.2009.)

[URI41] *Wuala*, http://www.wuala.com/ (12.9.2009.)
[URI42] *Yahoo*, http://www.yahoo.com/ (2.9.2009.)
[URI43] *YouTube*, http://www.youtube.com/ (17.7.2009.)
[URI44] *Zoho Creator*, http://creator.zoho.com/ (22.8.2009.)
[URI45] *Zoho Writer*, http://writer.zoho.com/ (22.8.2009.)

11 POPIS OZNAKA

11.1 Popis slika

Slika 2.1 Povijesni razvoj sustava za računalom podržano učenje 11
Slika 2.2 Struktura VLE sustava 19
Slika 3.1 Slojevi protokola SOA web servisa 23
Slika 3.2 Primjer korištenja XML-RPC protokola 24
Slika 3.3 Primjer razmjene poruka SOAP protokolom 25
Slika 3.4 REST arhitektura 26
Slika 3.5 Primjer REST komunikacije 27
Slika 3.6 Opis funkcioniranja RSS-a 29
Slika 3.7 Sinkroni model funkcioniranja web aplikacije 33
Slika 3.8 Asinkroni model funkcioniranja RIA 33
Slika 3.9 Proces provjere identiteta pokrenut od strane pružatelja usluga 37
Slika 3.10 Proces provjere identiteta pokrenut od strane davatelja elektroničkog identiteta 38
Slika 4.1 Konceptualni model PLE-a 47
Slika 4.2 Blog kao agregator sadržaja 53
Slika 4.3 Osobni web portal baziran na *Ajax* tehnologiji 54
Slika 4.4 Plan učenja s pridruženim resursima u mPLE sustavu 55
Slika 4.5 Osobni okoliš za učenje kreiran pomoću PLEF aplikacije 56
Slika 5.1 Model sustava 60
Slika 5.2 Dijagram slučajeva korištenja 61
Slika 5.3 Primjer konceptualne mape 70
Slika 5.4 Logički model podataka 73
Slika 6.1 Arhitektura aplikacije *Samouk* 75
Slika 6.2 UML dijagram najvažnijih klasa klijentskog dijela aplikacije *Samouk* 78
Slika 6.3 Načini komunikacije između *Adobe Flash* klijenta i poslužitelja 80
Slika 6.4 Prikaz procesa prijenosa podataka o materijalima 81
Slika 6.5 UML dijagram klasa paketa *Dokumenti* 84
Slika 6.6 UML dijagram klasa paketa *Biljeske* 86
Slika 6.7 Konceptualni model relacijske baze podataka 87

Slika 6.8 Proces provjere identiteta korištenjem *OpenID* protokola 89

Slika 6.9 Konceptualna mapa opisana CXL jezikom 93

Slika 7.1 Prijava u aplikaciju *Samouk* 97

Slika 7.2 Provjera identiteta korisnika aplikacije *Samouk* korištenjem *Yahooa* kao davatelja *OpenID* identiteta 98

Slika 7.3 Korisničko sučelje aplikacije *Samouk* 99

Slika 7.4 Stvaranje koncepta 100

Slika 7.5 Pridruživanje svojstava konceptu 101

Slika 7.6 Stvaranje vezanog koncepta 101

Slika 7.7 Pridruživanje svojstva vezi između koncepata 102

Slika 7.8 Pridruživanje resursa konceptu 103

Slika 7.9 Materijali vezani uz koncept 103

Slika 7.10 Uređivanje bilješki 104

Slika 7.11 Pretplata na RSS sadržaj 105

Slika 7.12 Tijek sinkrone komunikacije 105

Slika 7.13 Mapa koja predstavlja plan učenja 106

Slika 7.14 Mapa koja se koristi pri pisanju rada 108

11.2 Popis tablica

Tablica 4.1 Usporedba sustava za stvaranje PLE-a 57

Tablica 5.1 Funkcionalni zahtjevi vezani uz segment prijave korisnika 63

Tablica 5.2 Funkcionalni zahtjevi vezani uz segment administracije okruženja 64

Tablica 5.3 Funkcionalni zahtjevi vezani uz podsustav za podršku procesu učenja 65

Tablica 7.1 Usporedba aplikacije *Samouk* s drugim sustavima za stvaranje PLE-a 115

12 SAŽETAK

PROGRAMSKA POTPORA STVARANJU OSOBNOG OKOLIŠA ZA UČENJE

U ovom radu opisana je realizacija programske potpore stvaranju osobnog okoliša za učenje (engl. *Personal Learning Environment*, skraćeno PLE). PLE je okruženje koje učenik samostalno stvara, prilagođava i njime upravlja, što je posebno važno danas kada je dominantan oblik stjecanja znanja cjeloživotno samousmjereno učenje. U teorijskom dijelu rada napravljen je pregled tehnologija i sustava koji su utjecali na ideju oblikovanja PLE-a. Praktični dio rada obuhvaća opis modela sustava za stvaranje PLE-a i aplikacije *Samouk* realiziranu po tom modelu. Kreirana aplikacija omogućuje stvaranje PLE-a u kojem učenici predstavljaju svoje znanje u obliku konceptualnih mapa. Tijekom učenja, svakom elementu mape pridružuju materijale u elektroničkom obliku koje dobavljaju i pohranjuju korištenjem Web 2.0 servisa. Na takav način mapa postaje repozitorij u kojem su trajno pohranjene informacije o određenom području. Aplikacija je testirana u praksi, a na kraju rada su navedene preporuke i smjernice daljnjeg razvoja modela i aplikacije.

13 SUMMARY

SOFTWARE FOR CREATION OF PERSONAL LEARNING ENVIRONMENT

This thesis describes a model and its application for the creation of a Personal Learning Environment (PLE). PLE provides an opportunity for the user to create and manage his or her own environment for learning and knowledge exchange. It is very important because today lifelong self-directed learning has become increasingly frequent. Technologies and systems that influence the development of PLE are described in the theoretical part of the thesis. The practical part describes a software model for the creation of PLE and an application named *SelfTaught* (cro. *Samouk*). This application provides a simple interface that can be used to create a learning plan in the form of a concept map. During the learning process, the learner finds online content using Web 2.0 services, and links it to a particular concept. Filled with content, the concept map is transformed into a knowledge map and it becomes a permanent knowledge repository. The application is tested in practice, and in the final part of the thesis proposals and guidelines for the future development of a model and the *SelfTought* application are described.

14 KLJUČNE RIJEČI

Ključne riječi:

Osobni okoliš za učenje, PLE, elektroničko učenje, samousmjereno učenje, Web 2.0, društvene mreže, web servisi, RIA

Keywords:

Personal Learning Environment, PLE, e-learning, self-directed learning, Web 2.0, social networks, web services, RIA

15 ŽIVOTOPIS

Rođen sam 7. veljače 1972. godine u Otočcu gdje sam pohađao osnovnu i srednju školu. Nakon odsluženja vojnog roka, 1991. godine upisao sam Fakultet organizacije i informatike u Varaždinu, smjer Projektiranje informacijskih sustava. Diplomirao sam 1998. godine s temom „Upravljanje zakrčenjem u paketskim mrežama“. Od 1997. do 2008. godine radio sam u informatičkom poduzeću LAUS CC u Dubrovniku, gdje sam napredovao na radnim mjestima od programera do pomoćnika direktora razvoja. Na Odjelu za elektrotehniku i računarstvo Sveučilišta u Dubrovniku kao vanjski suradnik radim od 2004. godine kada sam upisao poslijediplomski magistarski studij računarstva na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Od 2006. godine stalno sam zaposlen kao asistent na Odjelu za elektrotehniku i računarstvo Sveučilišta u Dubrovniku, gdje sudjelujem u izvođenju nastave iz grupe računarskih predmeta. Područja mog znanstvenog i stručnog interesa su projektiranje i izgradnja informacijskih sustava i računalom podržano učenje. Autor sam jednog, a koautor tri znanstvena rada objavljena na međunarodnim znanstvenim konferencijama i u časopisima.