SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 324

On-line prikaz obogaćenih video snimki predavanja

Josip Petrić

Zagreb, srpanj 2011.

Sadržaj

Uvod

Cilj obrazovnih ustanova ali i svih predavača je ostvariti što kvalitetniji prijenos znanja na studente. Predavanja su oduvijek bila osnovni i najbolji način prenošenja znanja s predavača na slušatelja. Iako su dobra predavanja osnovni i najbolji alat kvalitetnog obrazovanja, ipak imaju svoja ograničenja. Neka ograničenja predavanja su: samo ograničeni broj slušatelja može prisustvovati predavanju te kada jednom završi identično predavanje se ne može više ponoviti. Kvaliteta obrazovanja znatno bi se povećala kada bi se dobra i kvalitetna predavanja mogla prikazati velikom broju zainteresiranih. Kvaliteta bi se također znatno povećala kada bi se ista predavanja mogla više puta odslušati. Rješenje ovih osnovnih nedostataka predavanja je jednostavno: predavanje treba snimiti. Zahvaljujući Internetu i globalnoj povezanosti, snimljeno predavanje se lako može prikazati svoj zainteresiranoj publici i neograničeno ponavljati. Iako snimka predavanja rješava navedene nedostatke predavanja, ima i vlastitih nedostataka. Odsutnost komunikacije između predavača i slušatelja je glavni nedostatak snimke u odnosu na predavanje. Mnoge obrazovne ustanove, ali i razvojne kompanije ulažu znatne napore kako bi snimke predavanja učinile što realnijim stvarnom predavanju.

Tema ovog rada je analizirati već postojeće alate te načine snimanja i prikaza predavanja. Na temelju analize već postojećih alata treba predložiti i izraditi jedinstveni alat za on-line prikaz predavanja.

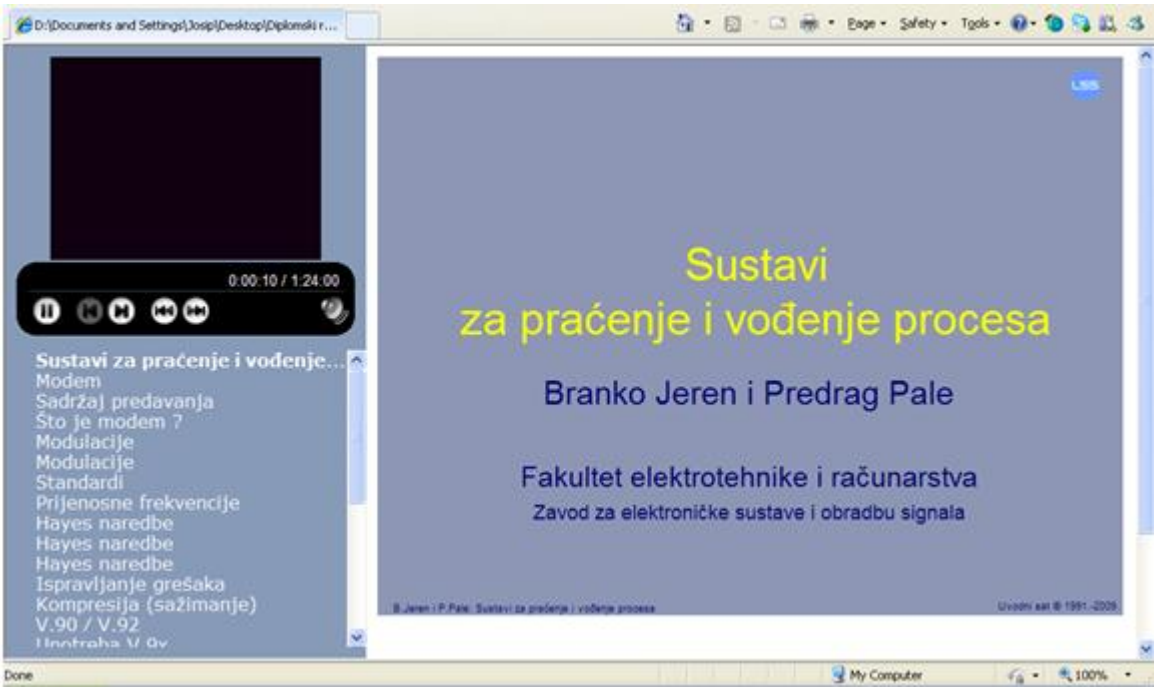
Predloženi i izrađeni alat nazvan je *Lecture Capturing Tool* ili skraćeno *LeCTo*. Alat LeCTo sadrži dva istovremena i sinkronizirana video sadržaja: video predavača i video prikazivanog sadržaja. Da bi snimka bila što realniji prikaz stvarnog predavanja, LeCTo omogućuje prikaz raznog dodatnog sadržaja. Dodatni sadržaj se lako stvara i dodaje snimci, a može biti: tekst, podatkovna poveznica, pitanja za provjeru znanja pa i cijele web stranice. Ono što LeCTo čini posebnim je mogućnost sinkronizacije dodatnog sadržaja sa snimkom tako da se omogućuje korisniku određivanje točnog trenutka kada će se neki dodatni sadržaj pojaviti, ali i nestati.

Ovaj dokument istaknut će posebnosti te prikazat detalje implementacije i rada alta LeCTo za on-line prikaz predavanja.

1. Postojeća rješenja

1.1. Microsoft Producer

Microsoft Producer je dodatak za *PowerPoint* prezentacijski alat. Alat se sastoji od dva dijela: dodatak za alat *PowerPoint* i samostalni alat pomoću kojeg se dodaje i sinkronizira dodatni sadržaj. Dodatak omogućuje sinkronizaciju slika te video i audio snimaka s prezentacijom kako bi se stvorila multimedijska prezentacija. Osnova multimedijske prezentacije je *PowerPoint* prezentacija. Multimedijska prezentacija je pogodna za prikaz putem web stranice [1].



Slika 1. Multimedijska prezentacija napravljena alatom MS Producer

Osnova kreiranja multimedijske prezentacije je *PowerPoint* prezentacija. Tijekom predavanja se posebnim dodatkom alata *PowerPoint* bilježe vremena promjene slajdova. Nakon što su sva vremena promjene slajdova zabilježena, koristi se alat *MS Producer* kako bi se prezentaciji dodao dodatni sadržaj te sinkronizirao sa snimkom. Prezentaciji se može dodati: audio i video snimke, slike te HTML stanice. Koristeći vremena promjene slajdova, svakom slajdu se može dodati određeni dodatni sadržaj (slika, audio ili video sadržaj). Nakon što je popratni sadržaj dodan i sinkroniziran, može se pokrenuti postupak objavljivanja (eng. *publish*) multimedijske

prezentacije. Objavljena prezentacija ima oblik web stranice. Glavne tehnologije korištene za prikaz multimedijske prezentacije su: *javascript* i *html*. Prilikom objavljivanja snimke, svaki slajd prezentacije je pretvoren u sliku JPG formata kako bi se lakše prikazivao u web pregledniku.

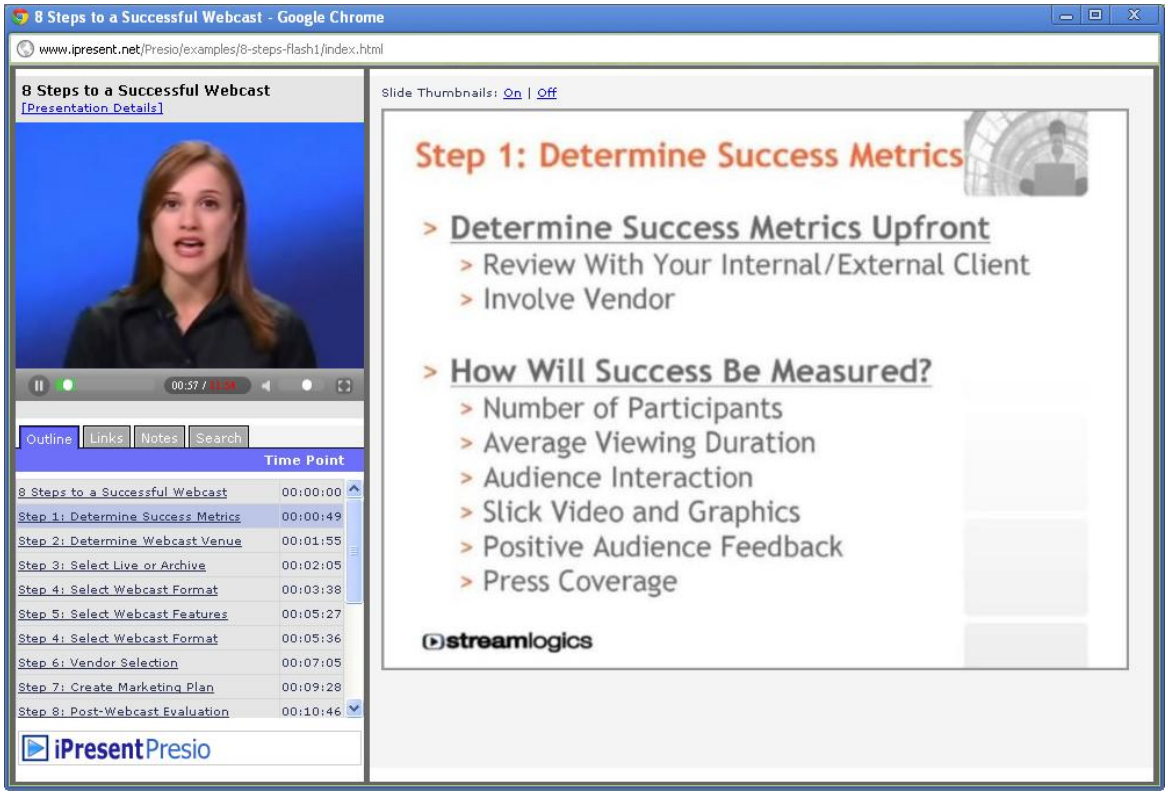
Iako alat *MS Producer* omogućuje lako i intuitivno dodavanje i sinkroniziranje raznog dodatnog sadržaja, ipak postoje nedostaci. Najveći nedostatak je mogućnost prikaza multimedijske prezentacije isključivo *Internet Explorer* i *Netscape Navigator* preglednikom. Orijentiranost samo na *PowerPoint* prezentaciju je također nedostatak jer ograničava broj potencijalnih korisnika. Multimedijalna prezentacija bila bi puno kvalitetnija kada bi se prezentaciji mogao dodati i popratni sadržaj poput: bilješki, podatkovnih poveznica, komentara i sličnih sadržaja.

1.2. Presio

Presio je samostalni alat koji omogućuje dodavanje i sinkroniziranje video ili audio snimke s *PowerPoint* prezentacijom. Alat je jednostavan za korištenje. Dodavanje i sinkroniziranje snimke obavlja se prateći šest jednostavnih koraka. U prvom koraku se upisuju osnovni podaci o prezentaciji: naziv, opis, datum prezentiranja, ime predavača te se mogu staviti neki dodaci prezentaciji. U drugom i trećem koraku se odabiru *PowerPoint* prezentacije i video snimke koje će biti obuhvaćene multimedijskom prezentacijom. Paralelno gledajući ili slušajući video ili audio snimku i prezentaciju korisnik odabire vremenske trenutke promjene slajdova. Na taj način se u četvrtom koraku obavlja sinkronizacija video ili audio snimke i *PowerPoint* prezentacije. U petom koraku moguće je pregledati napravljenu multimedijsku prezentaciju te se u šestom koraku stvorena prezentacija može objaviti. Prezentacija se može objaviti kao snimka u: *flash, windows media* ili *quick time* video formatu [2].

Alat *Presio* omogućuje i snimanje video i audio snimki. Ako se *Presio* koristi za snimanje video ili audio snimke predavanja tada nije potrebno provoditi šest navedenih koraka sinkronizacije jer alat *Presio* automatski sinkronizira snimku i prezentaciju.

Izgled multimedijalne prezentacije izrađene alatom *Presio* prikazan je na slici 2. *Presio* multimedijalna prezentacija sadrži: paralelni prikaz snimke predavanja i same prezentacije, navigacijske poveznice, podatkovne poveznice i bilješke povezane uz svaki slajd.

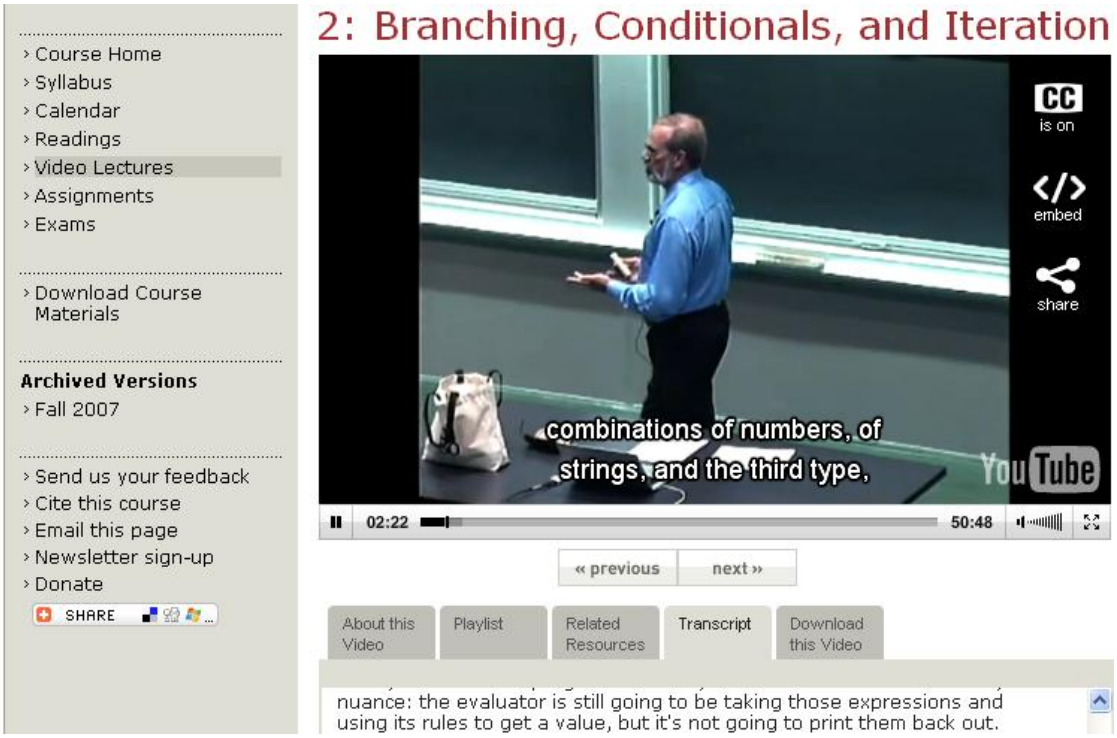


Slika 2. Prikaz multimedijske prezentacije stvorene alatom Presio
Slika preuzeta s weba: http://www.ipresent.net/Presio/examples.php

1.3. MIT OpenCourseWare

Kako bi se kvaliteta obrazovanja povećala, mnoge obrazovne ustanove razvijaju svoje vlastite sustave za e-learning. Najčešće su upravo snimke predavanja glavni dio tih e-learning sustava. Sveučilište MIT (*Massachusetts Institute of Technology*) svim zainteresiranim omogućuje pregled snimki predavanja. Snimke predavanja dio su njihovog e-learning sustava nazvanog *OpenCourseWare* [3]. Na slici 3 prikazan je primjer jedne snimke predavanja.

Za on-line prikaz predavanja, *MIT OpenCourseWare* sustav koristi jednostavne snimke predavanja bez popratne, sinkronizirane prezentacije ili dodatnog sadržaja. Iako su obične snimke predavanja, nemaju visokih tehnoloških zahtjeva te su lako dostupne svima. Dodavanje popratnog sadržaja znatno bi povećalo kvalitetu predavanja.



Slika 3. Primjer snimke predavanja *MIT OpenCourseWare* sustava
Slika preuzeta s weba: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/

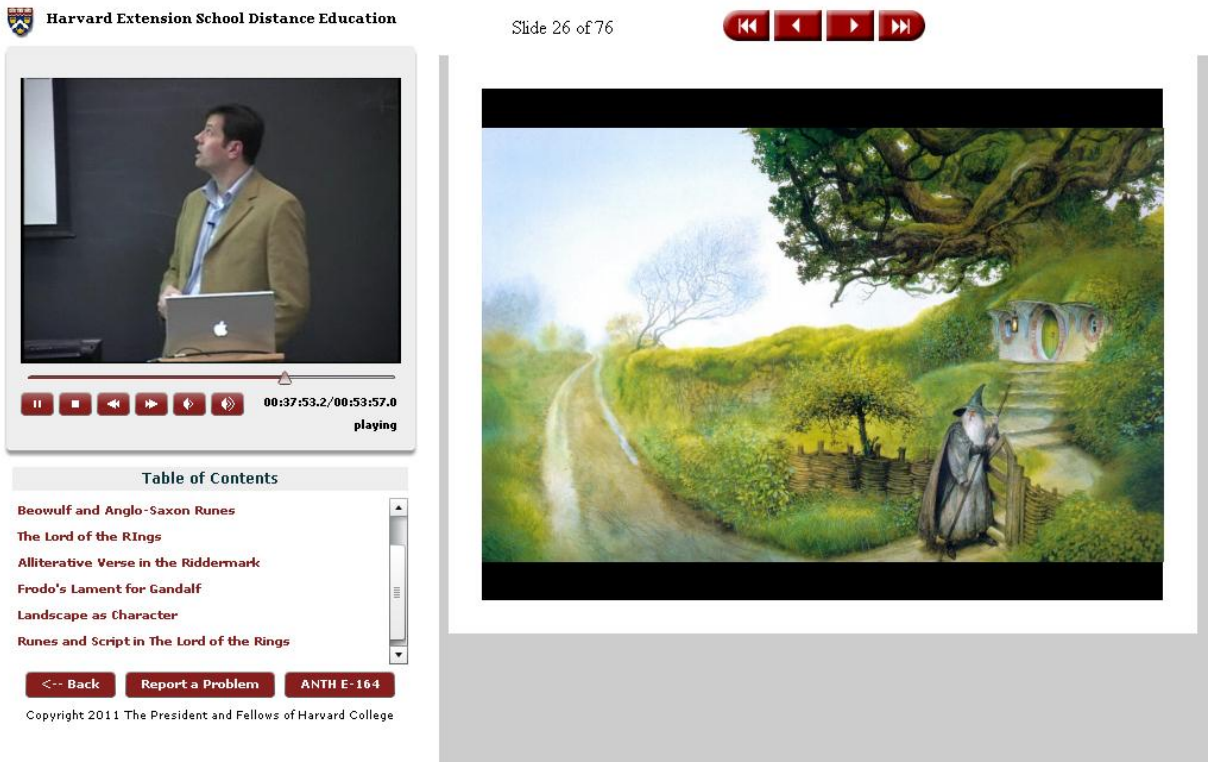
1.4. Harvard Extension Classroom

Sveučilište Harvard također omogućuje pregled predavanja putem multimedijalnih snimki predavanja. Multimedijalne snimke predavanja koje pruža sveučilište Harvard sastoji se od: video snimke predavača, prezentacije predavanja i sadržaja predavanja koji služi za navigaciju po snimci [4]. Prezentacija predavanja je sinkronizirana sa snimkom predavanja, stoga će se odabirom određenog slajda prezentacije snimka pomaknuti na dio gdje predavač govori o sadržaju s tog slajda. Nasuprot tome, ako se snimka pomakne na određeni dio koristeći navigacijsku traku alata za reprodukciju (eng. *player*[1]), prezentacija se neće pomaknuti na taj dio.

Harvard Extension Classroom za prikaz multimedijalne prezentacije predavanja koristi web tehnologije: *flash*, *html* i *javascript*. Player za reprodukciju snimke i navigacija po slajdovima prezentacije izrađeni su korištenjem *flash* tehnologije. Svaki slajd prezentacije pretvoren je u zasebni PDF dokument, što navigaciju čini znatno jednostavnijom. Kada korisnik odabere

[1] Eng. *Player* – alat za reprodukciju multimedijalnog sadržaja. U daljnjem tekstu će se koristiti riječ „player“ umjesto „alat za reprodukciju multimedijalnog sadržaja“.

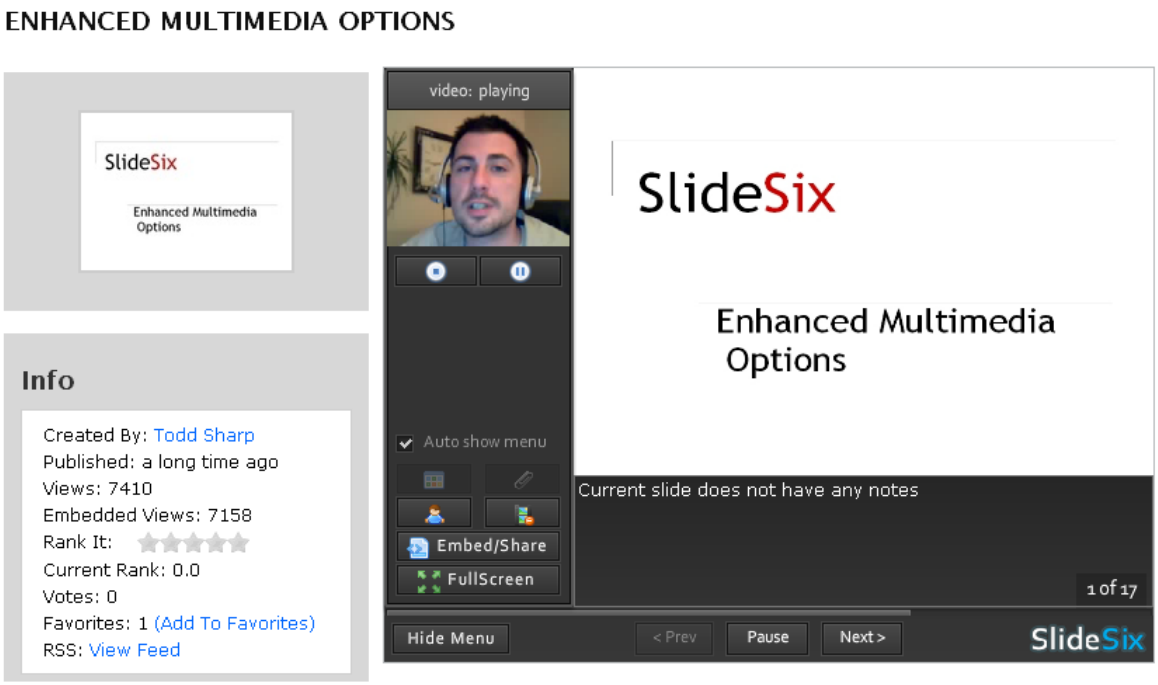
neki slajd, player počne reproducirati snimku za to vrijeme, a u posebni dio prozora se prikaže PDF dokument koji predstavlja traženi slajd.



Slika 4. Multimedijalna prezentacija *Harvard Extension Classroom* sustava
Slika preuzeta s weba: http://cm.dce.harvard.edu/2011/02/23228/L01/seg2/index_FlashSingleHighBandwidth.shtml

1.5. SlideSix

SlideSix je zajednica korisnika i besplatni sustav za izradu i dijeljenje prezentacija obogaćenih multimedijalnim sadržajem. Koristeći sustav svatko može učitati prezentaciju i snimku te ih sinkronizirati. Sustav također nudi alate za izradu prezentacije i snimanje video ili audio sadržaja koji se nakon snimanja automatski spaja s prezentacijom. Video ili audio snimka se može pridružiti cijeloj prezentaciji, ali i samo određenim slajdovima. Uz video ili audio snimku, korisnik ima mogućnost dodavanja bilješki za svaki slajd prezentacije. Velika prednost sustava je podržavanje širokog spektra formata prezentacija koje korisnici mogu učitati. Korisnici mogu učitati: *PowerPoint* ili *OpenOffice* prezentacije, PDF dokumente ili *QuickTime* video snimke [5].



Slika 5. Prikaz multimedijalne prezentacije korištenjem SlideSix sustava
Slika preuzeta s weba: http://slidesix.com/about

Kvaliteta multimedijalne prezentacije znatno bi se povećala kada bi korisnici imali mogućnost prikazivanja i sinkroniziranja različitog dodatnog sadržaja poput: podatkovnih poveznica, često postavljanih pitanja, prijevoda ili nekog drugog sadržaja.

2. Zadaci projekta

Zadatak projekta i diplomskog rada je izraditi alat za on-line prikaz obogaćenih snimki predavanja. Alat mora istodobno prikazivati najmanje dvije video snimke: snimku predavača i snimku predavanja koja je najčešće *PowerPoint* ili *OpenOffice* prezentacija. Dvije snimke moraju biti sinkronizirane za cijelo vrijeme njihovog reproduciranja. Video snimke predavanja moraju biti obogaćene dodatnim sadržajem koji je sinkroniziran sa snimkama predavanja. Poželjno je da dodatni sadržaj može biti bilo kojeg oblika te da je potpuno prilagodljiv željama korisnika, tj. autora multimedijske prezentacije. Neki od mogućih tipova dodatnog sadržaja su: podatkovne poveznice, bilješke, pitanja i ponuđeni odgovori, slike, dijagrami i slično. Korisnik mora imati mogućnost izbora vremenskih trenutaka pojavljivanja i nestajanja određenog dodatnog sadržaja kao i mogućnost povezivanja određenog sadržaja s određenim slajdom prezentacije. Uz dodatni i sinkronizirani sadržaj, snimka predavanja znatno dobiva na kvaliteti ako postoje jednostavni mehanizmi navigacije po snimci. Zbog toga alat treba omogućiti jednostavnu i intuitivnu navigaciju po snimci koristeći: navigacijsku traku playera, navigaciju po slajdovima prezentacije i navigaciju po ključnim riječima.

Kako alat ne bi bio tehnički zahtjevan za poslužiteljsku i klijentsku stranu, trebaju se koristiti što jednostavniji mehanizmi i tehnologije, po mogućnosti *open source* tehnologije. Alat mora raditi na većini današnjih web preglednika kao i na većini operacijskih sustava.

Zbog zahtjeva da se snimke predavanja mogu prikazivati na svim operacijskim sustavima, ali i zbog zahtjeva za sinkronizacijom dvije video snimke, sustav je ograničen na samo jedan format video snimki. Zbog ograničenja web tehnologija koje se koriste, korisnik može gledati snimke predavanja samo u on-line modu rada.

2.1. Snimke predavanja

Alat LeCTo[2] omogućuje on-line prikaz dvije video snimke. Najčešće su to: snimka predavača i snimka pripadajuće prezentacije. Dvije snimke su u svakom trenutku sinkronizirane, te će se navigacijom po snimci obadvije snimke istodobno pomicati na željeno mjesto. Alat LeCTo omogućuje više načina navigacije snimkom. Korisnik može upravljati snimkom koristeći:

[2] LeCTo – naziv izrađenog alata za on-line prikaz obogaćenih snimki predavanja.

standardnu navigacijsku traku *video player* alata, navigacijsku po osnovnih cjelina predavanja ili koristeći navigaciju po ključnim riječima permutiranog indeksa.

Kako je glavni zahtjev na snimke da se mogu pravilno reproducirati u svim modernim web preglednicima i operacijskim sustavima, kao format snimki odabran je format *Flash Video* (FLV). FLV format je odabran jer ga podržavaju svi moderni web pretraživači i operacijski sustavi. Format je široko prihvaćen te ga kao osnovni format za video snimke koriste poznati servisi kao: *YouTube, metacafe, Reuters.com* i drugi [6].

2.2. Obogaćivanje snimke dodatnim sadržajem

Reproduciranje dvije paralelne video snimke predavanja samo je osnova alata LeCTo. Ono što alat LeCTo čini drugačijim i inovativnim je obogaćivanje snimki predavanja dodatnim sadržajem. Dodatnim sadržajem ostvaruje se jedan način interakcije između predavača i slušatelja, interakcije koja se ne može ostvariti običnom snimkom predavanja. Alat LeCTo podržava četiri osnovna tipa dodatnog sadržaja: tekstualni sadržaj, podatkovne poveznice, pitanja s ponuđenim odgovorima i web sadržaj.

Dodatni sadržaj ima oblik posebno strukturirane XML datoteke. Datoteka s dodatnim sadržajem je samostalna, zasebna datoteka, tj. nije ugrađena u snimku predavanja. Svaka XML datoteka može sadržavati samo jedan tip dodatnog sadržaja, a jednoj snimci se može pridružiti neograničen broj datoteka s dodatnim sadržajem. Dodatni sadržaj se gledatelju prikazuje u posebnim prozorima, po jednom za svaki dodatni sadržaj. Autor dodatnog[3] sadržaja određuje kada će se i koji dio sadržaja prikazati gledatelju. Dva su načina određivanja vremenskih trenutaka u kojima će se pojaviti određeni dio dodatnog sadržaja: povezivanje određenog dijela sadržaja sa slajdom prezentacije i određivanje točnih vremenskih trenutaka (u sekundama) pojavljivanja i nestajanja sadržaja. Svi vremenski zapisi potrebni za sinkronizaciju dodatnog sadržaja sa snimkom nalaze se u istoj XML datoteci kao i dodatni sadržaj koji se sinkronizira.

[3] Radi se razlika između predavača (autora snimke predavanja) i autora dodatnog sadržaja jer dodatni sadržaj je odvojen od same snimke predavanja. Stoga, osim predavača, bilo koja osoba može biti autor dodatnog sadržaja.

2.2.1. Tekstualni dodatni sadržaj

Tekstualni dodatni sadržaj će se gledatelju prikazivati kao običan tekst, ali zbog dva moguća načina sinkronizacije sadržaja sa snimkom, taj sadržaj može biti vrlo raznolik. Autor dodatnog sadržaja može stvoriti sadržaj koji će se korisniku prikazivati zasebno za svaki slajd prezentacije, na primjer: predavačeve bilješke uz svaki slajd, često postavljana pitanja uz taj slajd, zanimljivosti vezane uz sadržaj tog slajda i slično. Autor dodatnog sadržaja može stvoriti sadržaj koji će se korisniku prikazivati u točno određenim vremenskim periodima, na primjer: prijevod predavačeva govora na druge jezike (eng. *subtitles*). Alat LeCTo ne postavlja druga ograničenja na sadržaj, osim da će gledatelju biti prikazan u obliku običnog teksta.

2.2.2. Podatkovne poveznice

Autor dodatnog sadržaja može referencirati gledatelje na dodatne sadržaje koristeći podatkovne poveznice. Prikaz podatkovnih poveznica se također može sinkronizirati sa snimkom na dva načina: povezivanjem poveznica s trenutnim slajdom ili određivanjem točnih vremenskih trenutaka.

2.2.3. Pitanja s ponuđenim odgovorima

Dodatni sadržaj u obliku pitanja s ponuđenim odgovorima omogućuje autoru stvaranje testova s ponuđenim odgovorima. U zadanim vremenskim trenutcima korisniku je sadržaj prikazan u obliku testa s ponuđenim odgovorima. Odabirom odgovora, gledatelj dobiva povratnu informaciju o točnosti odgovora. Autor sadržaja može odrediti vremenske trenutke pojavljivanja određenih testova. Koristeći ovaj tip dodatnog sadržaja studenti lako mogu provjeriti svoje znanje o sadržaju predavanja te tako dobiti povratnu informaciju o razumijevanju gradiva.

2.2.4. Web sadržaj

Najzanimljivija mogućnost koju autor dodatnog sadržaja ima je dodavanje cjelokupnih web stranica kao dodatnog sadržaja predavanju. Kao i ostali dodatni sadržaj, autor ima mogućnost odabira vremenskih trenutaka kada će se i koja web stranica učitati i prikazati korisnicima. Učitavanje i prikaz web stranice neće poremetiti on-line prikaz snimke jer će se željena web stranica učitati u posebnom prozoru. Na ovaj način autor dodatnog sadržaja ima potpunu

fleksibilnost kod izrade dodatnog sadržaja jer može izraditi vlastitu web stranicu s proizvoljnim sadržajem, na primjer: slike, druge video snimke, dijagrami, animacije i drugo.

2.3. Organizacija datoteka

Da bi uspješno reproducirao obogaćenu snimku predavanja, alat LeCTo koristi dvije vrste ulaznih datoteka: video snimke u FLV formatu i XML datoteke s konfiguracijskim parametrima i dodatnim sadržajem.

2.3.1. Video snimke

Alat LeCTo služi isključivo za prikaz obogaćenih video snimki predavanja, stoga LeCTo korisnicima ne omogućuje pohranu video snimki na poslužitelj. Korisnici mogu svoje video snimke pohraniti na bilo koji poslužitelj, ali snimka mora biti javno dostupna kako bi ju LeCTo Player[4] mogao dohvatiti. Da bi LeCTo Player mogao dohvatiti video snimku, korisnik mora u LeCTo Player učitati URL adresu snimke predavanja. URL adresa se učitava koristeći konfiguracijsku XML datoteku koja je jedna od ulaznih XML datoteka alata LeCTo.

Ovisno o poslužitelju na kojem je video snimka pohranjena, postoje dva načina na koje LeCTo Player reproducira snimke. Ako su snimke pohranjene na poslužitelj koji omogućuje *Real Time Messaging Protocol* (RTMP) [1] streaming tada će se korisnik moći kretati po onim dijelovima snimke koje alat LeCTo još nije učitao. Ako su snimke pohranjene na poslužitelj koji ne omogućuje RTMP streaming tada će se snimka prenositi HTTP protokolom pa se korisnik neće moći kretati po dijelu snimke koji još nije učitan.

2.3.2. Ulazne XML datoteke

Uz video snimke, kao ulazne datoteke potrebno je učitati i ulazne XML datoteke. Postoji tri vrste ulaznih XML datoteka: konfiguracijska XML datoteka (*Presentation Setup – PSU* datoteka), sinkronizacijska datoteka (*Content Change Event – CCE* datoteka) te datoteke s dodatnim sadržajem (*Additional Lecture Content – ALC* datoteka). Da bi LeCTo Player mogao reproducirati snimku predavanja potrebno je učitati konfiguracijsku (PSU) i sinkronizacijsku

[4] LeCTo Player – naziv aplikacije u sklopu LeCTo alata za reproduciranje obogaćenih snimki predavanja. LeCTo alat je alat s dvije glavne komponente: LeCTo Player i LeCTo Recorder. Ovaj rad bavi se komponentom LeCTo Player.

datoteku (CCE). Ako korisnik ne učita niti jednu datoteku s dodatnim sadržajem (ALC datoteku), LeCTo Player će reproducirati snimke predavanja koje neće biti obogaćene dodatnim sadržajem.

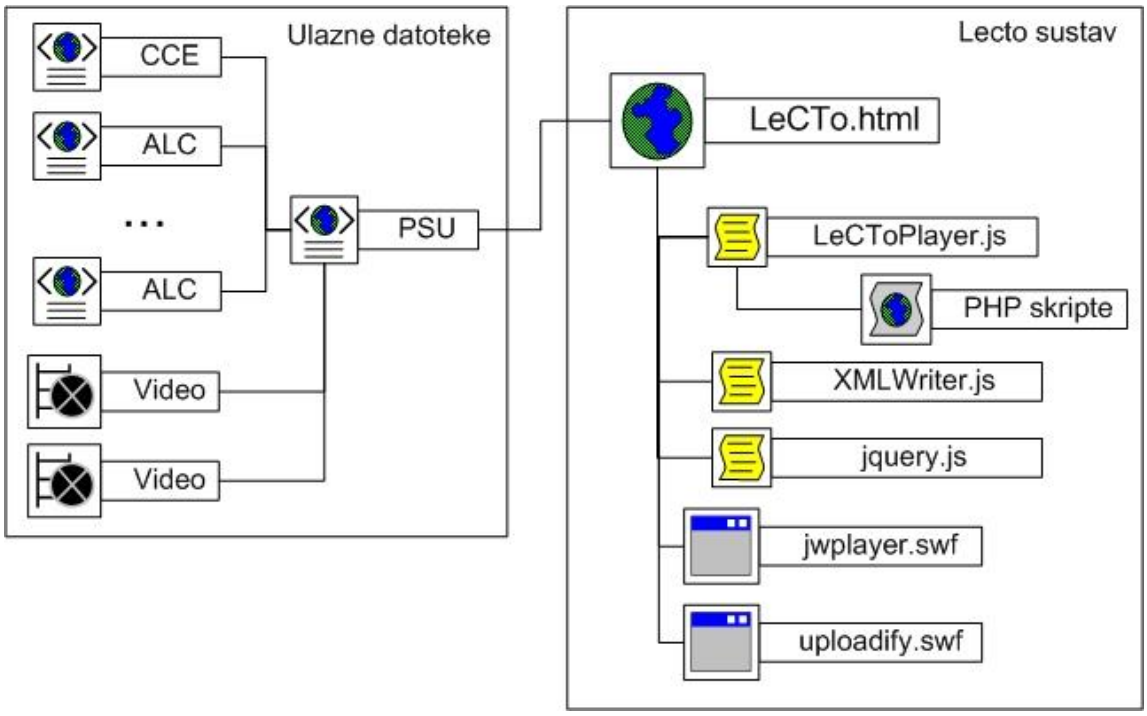
PSU konfiguracijska datoteka je osnovna datoteka alata LeCTo. PSU datoteka je XML datoteka koja sadrži sve informacije o snimci koje su potrebne alatu LeCTo da bi uspješno reproducirao snimku. Informacije koje PSU datoteka sadrži su: web lokacija (URL adresa) dvije video snimke, web lokacija sinkronizacijske datoteke i web lokacije svih datoteka s dodatnim sadržajima. Zbog toga što PSU datoteka sadrži sve informacije potrebne za pokretanje snimke dovoljno je samo PSU datoteku učitati u LeCTo Player kako bi se snimka mogla reproducirati.

CCE sinkronizacijska datoteka sadrži zapise o vremenskim trenutcima promjene slajda prezentacije. Koristeći sadržaj sinkronizacijske datoteke dodatni sadržaj se sinkronizira s prezentacijom. Sinkronizacijska datoteka se koristi i za stvaranje navigacijskog izbornika s kojim korisnik lako može prebacivati snimku na željeni slajd. Ako je web lokacija CCE datoteke navedena u PSU konfiguracijskoj datoteci, tada korisnik ne mora zasebno učitavati CCE datoteku u LeCTo Player.

ALC datoteka je datoteka s dodatnim sadržajem koja služi za obogaćivanje snimke predavanja. ALC datoteka se sastoji od niza parova: sadržaj i vremenski trenutak kada će sadržaj biti prikazan korisniku. Učitavanje ALC datoteke u LeCTo Player nije uvjet za uspješno reproduciranje snimke. Korisnik sam odlučuje želi li uz snimku vidjeti i dodatni sadržaj ili ne.

LeCTo Player omogućuje dva načina učitavanja XML datoteka: navodeći URL adresu datoteke ili učitavanje datoteke izravno s lokalnog diska klijentskog računala.

3. Arhitektura sustava



Slika 6. Arhitektura LeCTo sustava

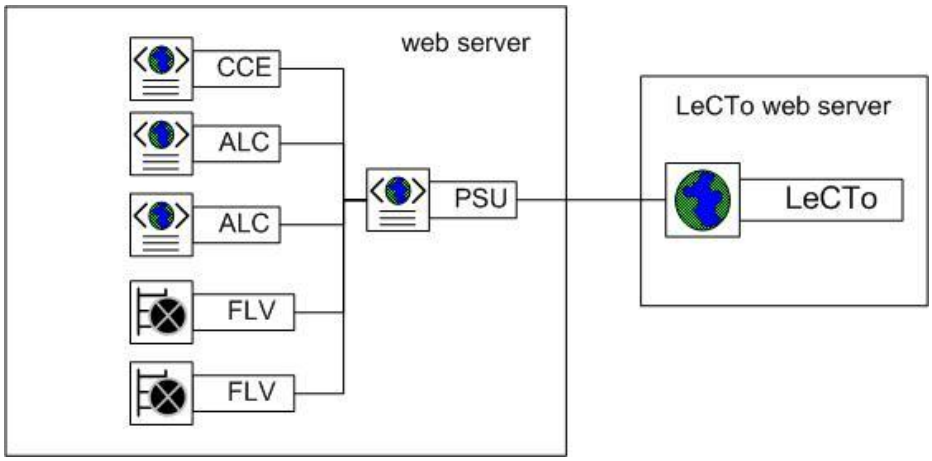
Na slici 6 prikazana je struktura LeCTo sustava. LeCTo sustav je osmišljen tako da se većina operacija obavlja na klijentskom računalu kako bi se što manje opteretio web poslužitelj. Zbog toga je sva pozadinska logika LeCTo sustava implementirana koristeći *javascript*$_2$ skripte. Iako se sva pozadinska logika obavlja na klijentskom računalu, neke je operacije potrebno obaviti na web poslužitelju. U tu svrhu se koriste PHP$_3$ poslužiteljske skripte koje se pozivaju iz *javascript* koda. Uz *javascript* i PHP skripte, LeCTo Player koristi gotove komponente: *jwplayer*$_4$ i *uploadify*$_5$. *Jwplayer* je multimedijalni *player* koji podržava *flash video* format. *Jwplayer* komponenta nudi laku integraciju u web stranicu, kao i *javascript* API[5] za lako integriranje s pozadinskom logikom LeCTo sustava. *Uploadify* je nadogradnja za *jquery* koja unapređuje formu za učitavanje datoteka (eng. *upload form*) tako što omogućuje učitavanje

[5] API – *Application Programming Interface* – točno određeni skup pravila i specifikacija koje drugi programi mogu pratiti kako bi međusobno komunicirali [7]. U programskom inženjerstvu je to obično skup metoda ili funkcija neke komponente koje se mogu pozivati iz drugih programa ili komponenti kako bi komponente međusobno komunicirale.

više datoteka na poslužitelj odjednom. Koristi se kada korisnik učitava ulazne datoteke putem forme za učitavanje datoteka.

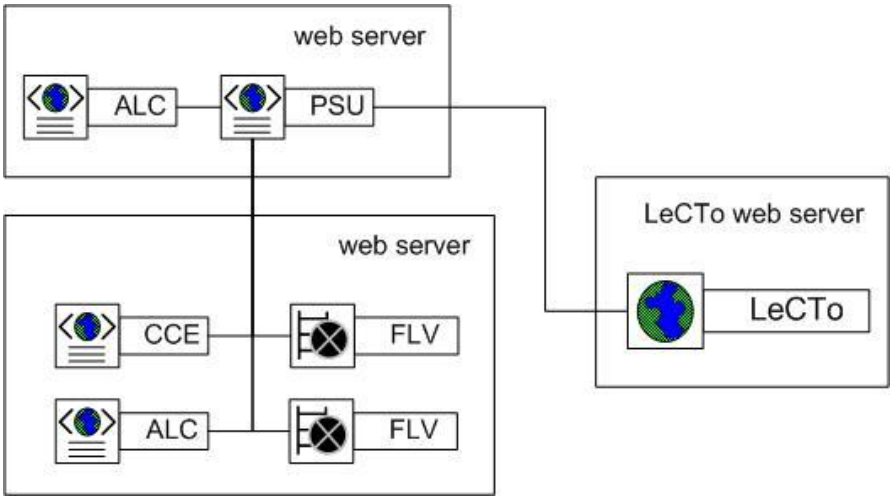
3.1. Razmještaj ulaznih datoteka

Ulazne datoteke moraju biti učitane u LeCTo sustav, kako bi se obogaćena snimka predavanja mogla reproducirati. Ulazne datoteke su: PSU konfiguracijska datoteka, CCE sinkronizacijska datoteka, ALC datoteke s dodatnim sadržajem i dvije video snimke. ALC datoteke s dodatnim sadržajem mogu biti izostavljene. Web lokacije svih potrebnih datoteka mogu biti navede u PSU konfiguracijskoj datoteci. Na taj način korisnik ne mora učitavati sve ulazne datoteke zasebno, već samo jednu datoteku, PSU datoteku. Kako alat LeCTo Player ne omogućuje stvaranje repozitorija s datotekama na vlastitom web poslužitelju, konfiguracijske datoteke i video snimke korisnik može pohraniti na razne web poslužitelje. Jedini zahtjev je da snimke budu javno dostupne kako bi ih LeCTo Player mogao dohvatiti.



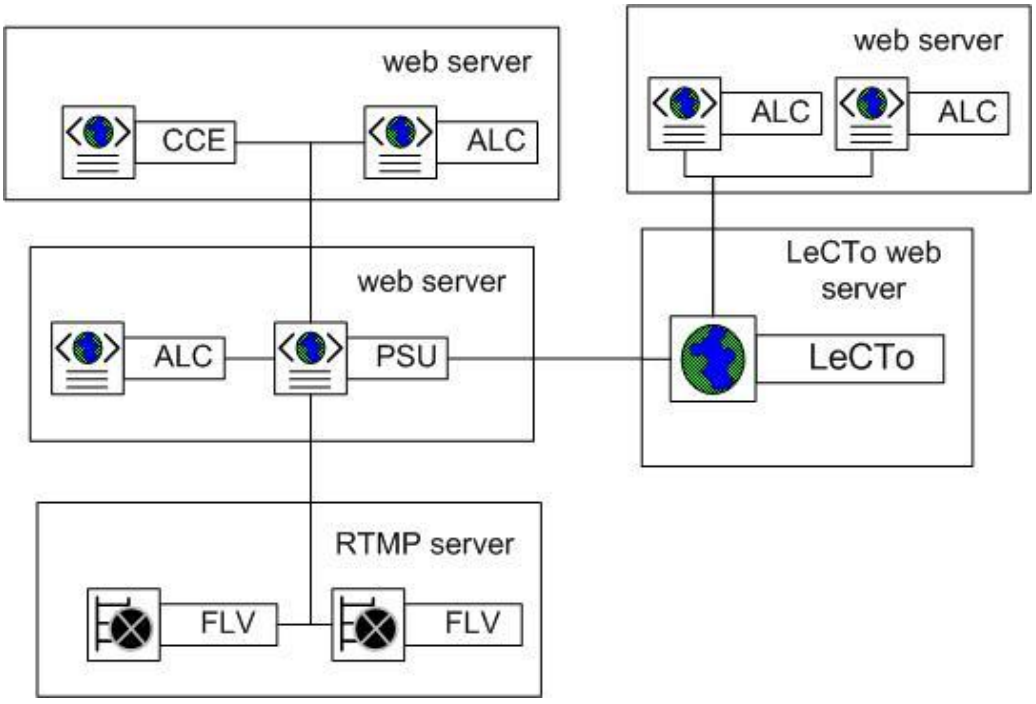
Slika 7. Primjer razmještaja ulaznih datoteka

Slika 7 prikazuje jedan od mogućih razmještaja ulaznih datoteka. Sve ulazne datoteke mogu biti smještene na web poslužitelj koji nije u istoj domeni kao LeCTo web poslužitelj. U prikazanom primjeru je PSU datoteka na istom poslužitelju kao i CCE, ALC i video datoteke. U PSU datoteci su navedene web lokacije (URL adrese) svih ostalih datoteka. Korisnik u alat LeCTo Player učitava samo PSU datoteku, a alat LeCTo na temelju navedenih web lokacija u PSU datoteci automatski dohvaća ostale datoteke.



Slika 8. Primjer razmještaja ulaznih datoteka

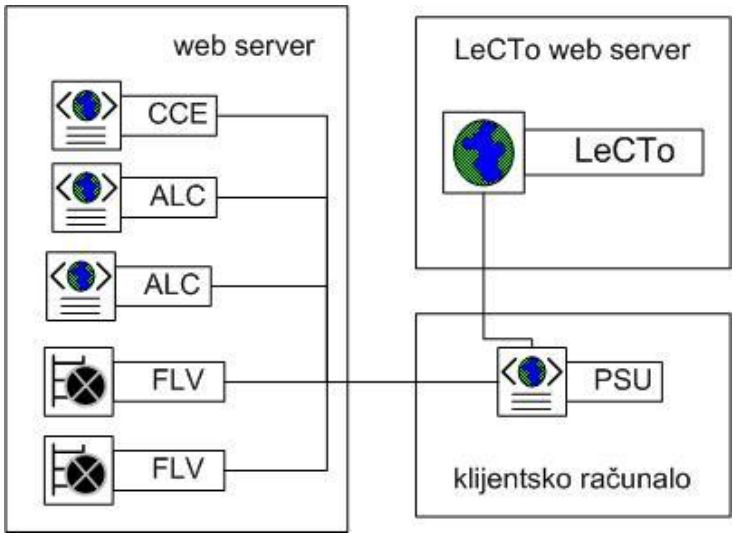
Slika 8 prikazuje primjer razmještaja ulaznih datoteka u kojem su datoteke razmještene na više poslužitelja. Svaki poslužitelj može biti u različitoj domeni. U ovom primjeru se samo PSU datoteka učitava u LeCTo Player. Kako su u PSU datoteci navedene web lokacije do svih ostalih datoteka, LeCTo Player automatski učitava sve navedene datoteke.



Slika 9. Primjer razmještaja ulaznih datoteka

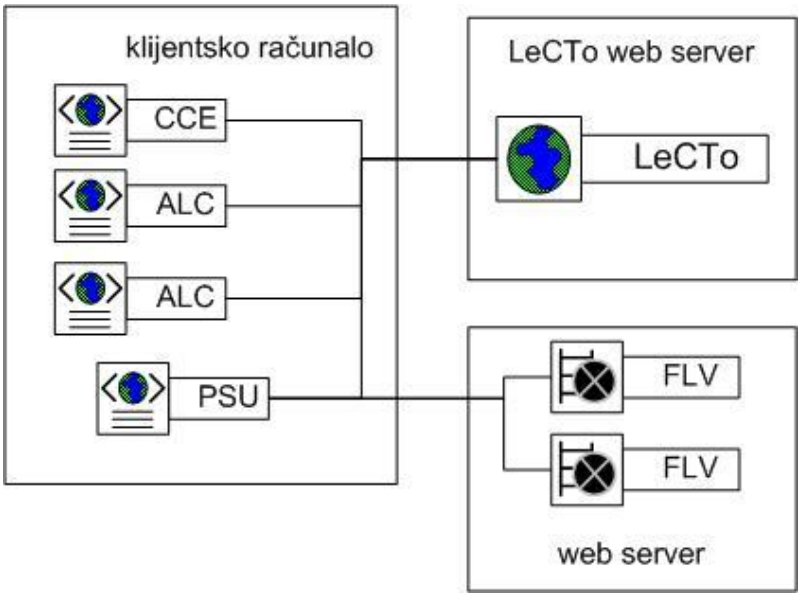
Slika 9 prikazuje dvije dodatne mogućnosti alata LeCTo. Uz PSU datoteku, u LeCTo Player se mogu izravno učitati dodatne ALC datoteke, bez da se navode u PSU datoteci. Ova mogućnost omogućuje korisnicima učitavanje dodatnih ALC datoteka bez potrebe prepravljanja PSU datoteke. Ako gledatelji snimke žele dodati još neki sadržaj snimci, a nemaju pristup PSU datoteci kako bi dodali nove ALC datoteke snimci, ova mogućnost im to omogućuje.

Ako postoji dostupan poslužitelj s omogućenim RTMP protokolom tada korisnik može video snimke pohraniti na takvom poslužitelju. Ako su snimke pohranjene na takvom poslužitelju tada će biti moguće streamanje video snimke. U suprotnom, ako su video snimke pohranjene na poslužitelju bez mogućnosti RTMP protokola tada streamanje video snimke neće biti moguće.



Slika 10. Primjer razmještaja ulaznih datoteka

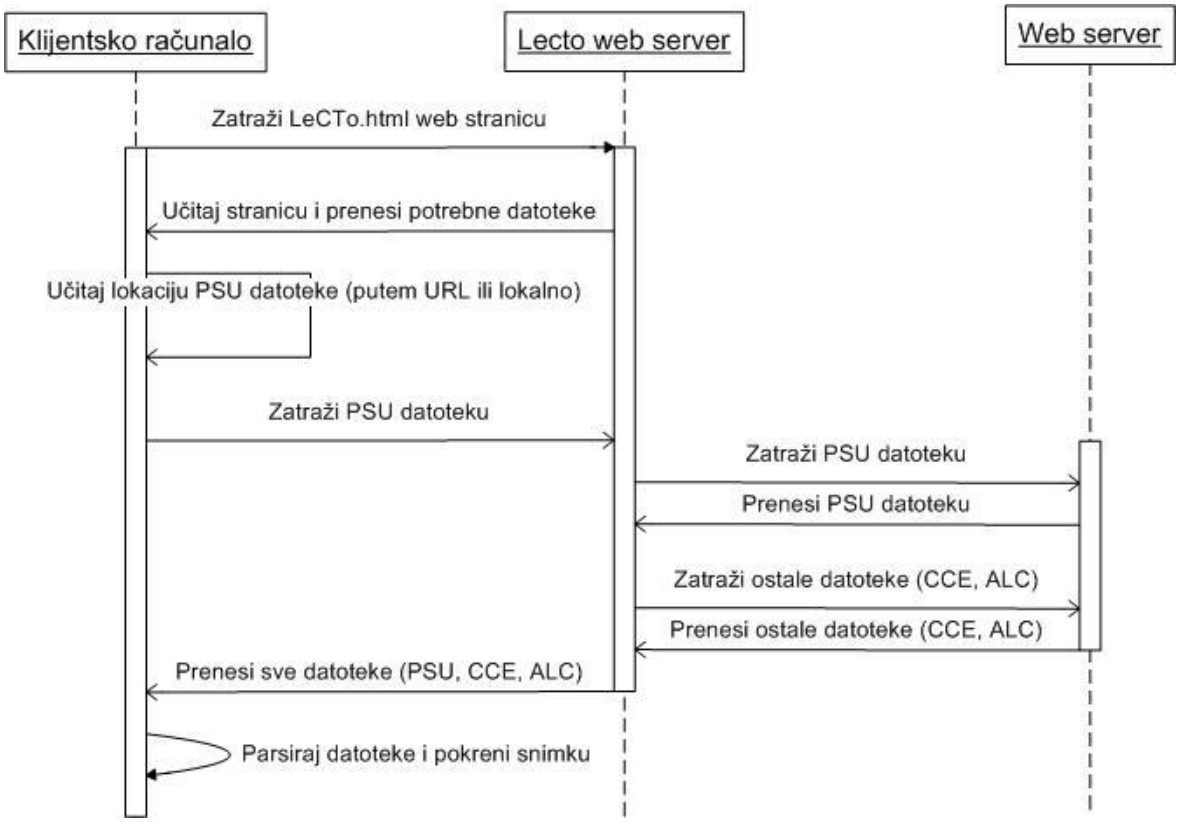
Slika 10 prikazuje primjer razmještaja ulaznih datoteka gdje se PSU datoteka nalazi na klijentskom računalu. Korisnik učitava PSU datoteku u LeCTo Player izravno sa svog računala. U PSU datoteci navedene su web lokacije svih ostalih potrebnih datoteka pa LeCTo Player automatski učitava ostale datoteke. Jedini zahtjev je da se video datoteke nalaze na nekom poslužitelju. Ako se sve potrebne XML datoteke učitavaju s klijentskog računala tada nije potrebno upisivati lokacije tih datoteka u PSU datoteci. U PSU datoteci je potrebno upisati lokaciju video snimki. Primjer kada se sve XML datoteke učitavaju s klijentskog računala prikazan je na slici 11.



Slika 11. Primjer razmještaja ulaznih datoteka

3.2. Općeniti način rada alata LeCTo Player

LeCTo Player je web aplikacija koja se nalazi na posebnom web poslužitelju. Kada korisnik pokrene LeCTo Player, tj. učita *LeCTo.html* datoteku, neke komponente se s poslužitelja prenesu na klijentsko računalo. Komponente koje se prenesu su: učitana HTML stranica, sve potrebne *javascript* datoteke i potrebne *css* datoteke. Javascript datoteke sadrže većinu programske logike LeCTo Player alata, jer je LeCTo Player izrađen tako da se većina zahtjevnih operacija obavlja na klijentu. Kada su sve potrebne datoteke prenesene na klijentsko računalo, korisnik može početi raditi s alatom LeCTo Player. Da bi se obogaćena snimka predavanja pokrenula, korisnik mora učitati barem PSU konfiguracijsku datoteku. U PSU datoteci trebaju biti navedene lokacije ostalih datoteka. Da bi LeCTo Player trebao raditi u PSU datoteci moraju biti navedene lokacije: CCE datoteke i lokacije dvije video snimke. Lokacije ALC datoteka nisu obvezne. Korisnik ima dvije mogućnosti učitavanja PSU datoteke u LeCTo Player: putem URL adrese ili izravno s lokalnog diska računala. Kada LeCTo Player učita PSU datoteku iz nje pročita URL adrese ostalih datoteka te ih automatski učita, bez potrebe korisničke intervencije. Kada su sve potrebne datoteke učitane, LeCTo Player može pokrenuti snimku. Sekvencijalni dijagram navedenih operacija prikazan je na slici 12.



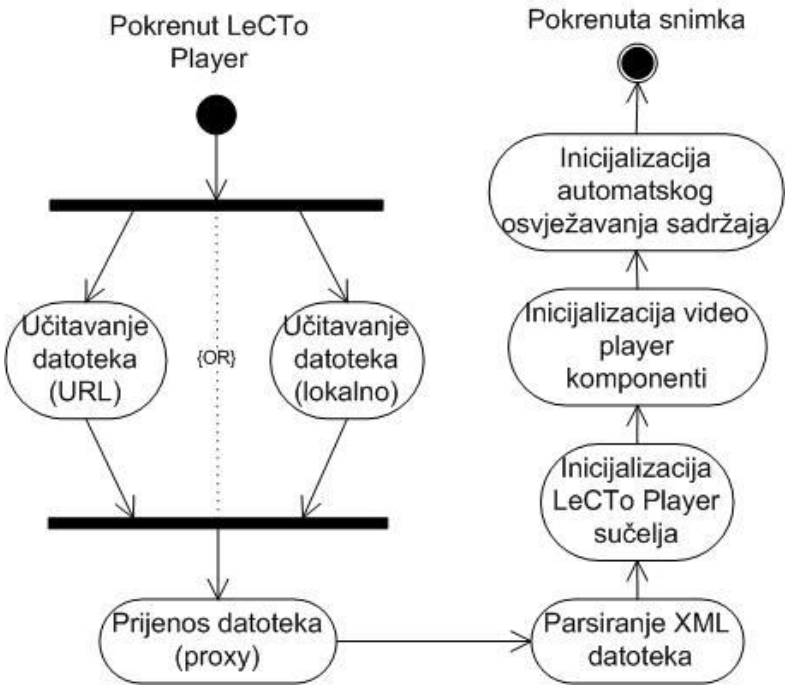
Slika 12. Sekvencijalni dijagram općeg rada LeCTo Player alata

Video datoteke se ne prenose na klijentsko računalo kao ostale datoteke. Video datoteke se prenose za vrijeme reproduciranja snimke, tj. korisnik ne mora čekati da se cijele video datoteke prenesu na klijentsko računalo.

4. Implementacija

Implementacija alata LeCTo Player dijeli se na četiri osnovne cjeline:

- upravljanje ulaznim datotekama i XML parsiranje
- inicijalizacija i prikaz LeCTo Player sučelja
- sinkronizacija i prikaz video snimki
- sinkronizacija dodatnog sadržaja sa snimkom



Slika 13. Dijagram aktivnosti LeCTo Player alata

Na slici 13 prikazan je slijed aktivnosti prilikom pokretanja snimke alatom LeCTo Player. Reproduciranje snimke počinje od učitavanja PSU konfiguracijske datoteke u LeCTo Player. PSU datoteka može biti učitana na dva načina: upisivanjem web lokacije PSU datoteke ukoliko se PSU datoteka nalazi na nekom poslužitelju ili učitavanjem datoteke s lokalnog diska. Uz PSU datoteku korisnik može učitati i dodatne ALC datoteke. Zbog toga što se parsiranje XML datoteka odvija na klijentskom računalu, sve XML datoteke se moraju moći pročitati s klijentskog računala. Posebnim mehanizmima prijenosa datoteke se prenesu s

poslužitelja na klijentsko računalo. Mehanizmi prijenosa su detaljno opisani u narednim poglavljima. Kada su datoteke učitane na klijentsko računalo, mogu se parsirati. Sadržaj datoteka sprema se u podatkovni model alata LeCTo Player. Na temelju učitanih podataka o snimkama i dodatnom sadržaju, priprema se grafičko sučelje i video player komponente alata LeCTo Player. Nakon što je grafičko sučelje inicijalizirano, pokreće se funkcija koja će automatski osvježavati i sinkronizirati dodatni sadržaj snimke.

4.1. Upravljanje ulaznim datotekama i XML parsiranje

Svi podaci o jednoj snimci nalaze se u XML datotekama. Tri su tipa XML ulaznih datoteka: PSU, CCE i ALC datoteke. Da bi LeCTo Player mogao čitati ulazne datoteke, one moraju imati točno određenu strukturu. Svaka vrsta ulaznih datoteka ima drugačiju strukturu.

4.1.1. Struktura PSU datoteke

PSU (*Presentation SetUp*) datoteka je konfiguracijska datoteka alata LeCTo Player. Svaka snimka može imati samo jednu PSU datoteku. U PSU datoteci se nalaze URL adrese svih ostalih datoteka: dvije video snimke, CCE datoteke i ako je korisnik naveo, adrese ALC datoteka. Uz adrese datoteka, navedene su i informacije o načinu kako će određeni sadržaj biti prikazan korisniku.

```
<?xml version="1.0"?>
<psu>
	<video>
		<window>
			<title>Lecture Video</title>
			<size>
				<width>400</width>
				<height>350</height>
			</size>
			<resizable>true</resizable>
			<location>
				<positionX>50</positionX>
				<positionY>50</positionY>
			</location>
		</window>
		<uri>http://diana.zesoi.fer.hr/~jpetric/videos/desktop.flv</uri>
	</video>
```

```
	<video>
		<window>
			...
		</window>
		<uri>http://diana.zesoi.fer.hr/~jpetric/videos/lecturer.flv</uri>
	</video>
	<cce>
	   <uri>http://weblectures.yolasite.com/resources/lecture.cce.xml</uri>
	</cce>
	<alc>
		<window>
			...
		</window>
	<uri>http://weblectures.yolasite.com/resources/SpeakerNotes.alc</uri>
	</alc>
	<alc>
		<window>
			...
		</window>
		<uri>http://weblectures.yolasite.com/resources/links.alc</uri>
	</alc>
	<alc>
		<window>
			...
		</window>

	<uri>http://weblectures.yolasite.com/resources/webSlides.alc</uri>
	</alc>
	<navigation>
		<window>
			...
		</window>
	</navigation>
</psu>
```

Slika 14. Struktura PSU ulazne datoteke

Slika 14 prikazuje strukturu PSU datoteke. PSU datoteka mora imati korijenski element *psu*. Četiri osnovna elementa PSU datoteke su: *video, cce, alc* i *navigation* element. Svi elementi osim *navigation* elementa sadrže i *window* element. *Window* element sadrži konfiguracijske podatke o prozoru alata LeCTo Player u kojem će biti prikazan pripadajući sadržaj. *Title* element sadrži naziv prozora. *Size* element visinu i širinu prozora. *Resizable* element označava smije li korisnik prozoru mijenjati veličinu. *Location* element označava lokaciju na kojoj će prozor biti prikazan iz Internet preglednika.

Jedna PSU datoteka može imati: samo dva *video* elementa, jedan *cce* element, jedan *navigation* element i neograničeno mnogo *alc* elemenata. *Video* elementi sadrže URL adrese video datoteka koje LeCTo Player treba reproducirati (*url* element). *Cce* element sadrži URL adresu CCE datoteke koja pripada snimci. *Alc* elementi sadrže URL adrese ALC datoteka s dodatnim sadržajima za snimku. *Navigation* element sadrži informacije kako će biti prikazan prozor s navigacijskim indeksom.

PSU datoteka mora imati naziv točno određenog formata: *naziv.cce.xml*.

4.1.2. Struktura CCE datoteke

CCE datoteka (*Content Change Events*) sadrži vremenske trenutke promjene sadržaja predavanja. Sadržaj predavanja može biti slajd *PowerPoint* ili *OpenOffice* prezentacije, PDF stranica ili neki drugi sadržaj. Vremenski trenuci promjene sadržaja su osnovne informacije potrebne za sinkronizaciju dodatnog sadržaja sa snimkom, ali i za izradu navigacijskog indeksa.

```
<?xml version="1.0"?>
<presentation>
	<frame>
		<contextID>257</contextID>
		<title>LeCTo Player</title>
		<occurrences>
			<occurrence>20.445</occurrence>
		</occurrences>
	</frame>
	<frame>
		<contextID>258</contextID>
		<title>Sinkronizacija sadržaja</title>
		<occurrences>
			<occurrence>50.7669</occurrence>
			<occurrence>98.871</occurrence>
		</occurrences>
	</frame>
	   ...
</presentation>
```

Slika 15. Struktura CCE ulazne datoteke

Slika 15 prikazuje strukturu CCE datoteke. CCE datoteka mora ima korijenski element *presentation*. Osnovni elementi CCE datoteke su *frame* elementi. Jedna datoteka može imati

neograničen broj *frame* elemenata. Najčešće je broj *frame* elemenata jednak broju slajdova prezentacije. *Frame* element sadrži: *contextID, title* i *occurrences* elemente. *ContextID* je jedinstveni identifikator koji označava jedan *frame* element. *Title* je naziv sadržaja, najčešće naziv slajda prezentacije. *Occurrences* element može sadržavati jedan ili više *occurence* elemenata. Svaki *occurence* element označava jedan vremenski trenutak kada se sadržaj treba prikazati gledatelju. U primjeru *PowerPoint* prezentacije, jedan *occurence* element označava vremenski trenutak kada se pojavio slajd prezentacije koji ima ID i naziv navedene u *contextID* i *title* elementima. Kako se jedan slajd može pojaviti više puta, jedan *frame* može imati više *occurrence* elemenata.

4.1.3. Struktura ALC datoteke

ALC datoteka sadrži dodatni sadržaj kojim se obogaćuje snimka predavanja. Datoteka sadrži sadržaj koji treba prikazati gledatelju te vremenske trenutak kada taj sadržaj treba biti prikazan te kada treba nestati. Vremenski trenutci prikazivanja i nestajanja sadržaja mogu biti iskazani na dva načina: točnim vremenskim trenucima ili navodeći vrijednost *contextID* sadržaja iz CCE datoteke. Navodeći vrijednost *contextID* sadržaja iz CCE datoteke povezujemo sadržaj sa vremenskim trenucima pojavljivanja željenog slajda. Postoji četiri varijacije ALC datoteke: ALC datoteka sa tekstualnim sadržajem, s podatkovnim poveznicama, s pitanjima i odgovorima te s web stranicama kao dodatnim sadržajem.

```
<?xml version="1.0"?>
<alc type="other">
	<window>
		<title>Lecture FAQs</title>
		<size>
			<width>250</width>
			<height>100</height>
		</size>
		<resizable>true</resizable>
		<location>
			<positionX>300</positionX>
			<positionY>200</positionY>
		</location>
	</window>
```

```
<frame>
	<occurrence>3.00</occurrence>
		<duration>20</duration>
		<content>
			FAQ pitanje za prvi slajd?
			Odgovor na postavljeno pitanje za prvi slajd.
		</content>
	</frame>
	<frame>
		<contextID>256</contextID>
		<content>
			Ovo je FAQ pitanje za drugi slajd prezentacije?
			FAQ odgovor za drugi slajd prezentacije.
		</content>
	</frame>
	  ...
</alc>
```

Slika 16. Struktura ALC ulazne datoteke

Slika 16 prikazuje strukturu ALC datoteke. ALC datoteka mora imati korijenski element *alc*. Osnovni element je *frame*. *Frame* element sadrži *content* element koji sadrži sadržaj koji treba prikazati gledatelju. Svaki sadržaj treba prikazati u određenom vremenskom periodu. Vremenski period se može iskazati na dva načina: povezivanjem sa željenim slajdom prezentacije pomoću *contextID* elementa ili određivanjem vremena početka (*occurrence* element) i vremena trajanja (*duration* element) u sekundama. Uz *frame* elemente, na početku ALC datoteke se nalazi *window* element koji sadrži postavke prozora u kojemu će biti prikazan dodatni sadržaj gledatelju. *Window* element u PSU datoteci će nadjačati *window* element za istu ALC datoteku.

Korisnik može izraditi četiri vrste dodatnog sadržaja: tekstualni sadržaj, podatkovne poveznice, pitanja i odgovori te web stranice. U ovisnosti o vrsti dodatnog sadržaja, mijenjat će se i struktura *content* elementa kao i atribut *type alc* elementa. Tipovi *type* atributa su: *others* (za tekstualni sadržaj), *web* (za web stranice), *links* (za podatkovne poveznice) i *quiz* (za pitanja i odgovore). Na slici 16 prikazan je primjer ALC datoteke s tekstualnim dodatnim sadržajem. Varijacije *content* elementa prikazane su na sljedećim slikama.

```
<?xml version="1.0"?>
<alc type="web">
	<frame>
		<contextID>256</contextID>
		<content>
			http://www.24sata.hr/
		</content>
	</frame>
	  ...
</alc>
```

Slika 17. ALC datoteka s podatkovnim poveznicama kao dodatnim sadržajem

Alc datoteka koja sadrži podatkovne poveznice može imati više *link* elementa unutar *content* elementa. *Link* element sadrži: *linkTitle* i *linkTo* elemente. *LinkTitle* sadrži naziv podatkovne poveznice koji će biti vidljiv korisniku. *LinkTo* sadrži podatkovnu poveznicu.

```
<?xml version="1.0"?>
<alc type="links">
	<frame>
		<contextID>256</contextID>
		<content>
			<link>
				<linkTitle>link</linkTitle>
				<linkTo>www.linik.com</linkTo>
			</link>
			<link>
				<linkTitle>drugi link</linkTitle>
				<linkTo>nekidrugilink.com</linkTo>
			</link>
		</content>
	</frame>
	  ...
</alc>
```

Slika 18. ALC datoteka koja sadrži web stranicu kao dodatni sadržaj

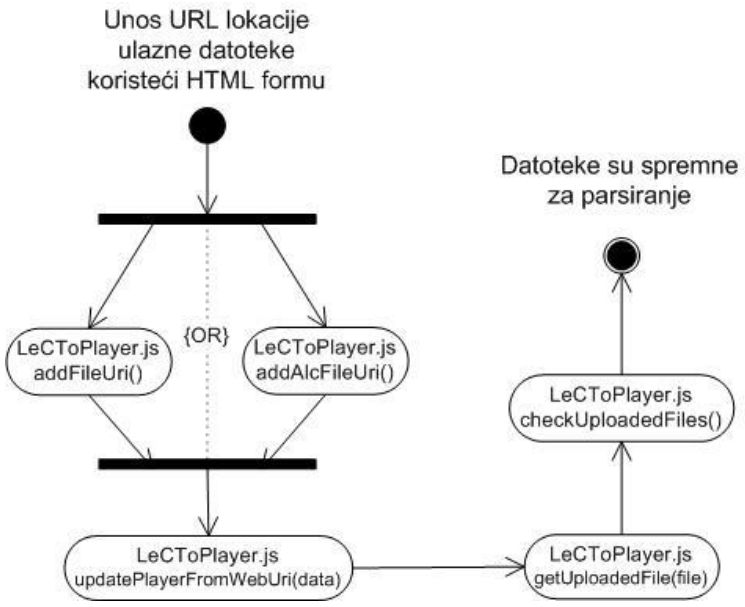
```
<?xml version="1.0"?>
<alc type="quiz">
      <frame>
            <contextID>257</contextID>
            <content>
                  <question>
                        <questionText>
                              Koliko paralelnih videa možemo prikazati
                              LeCTo Playerom?
                        </questionText>
                        <answer correct="false">1</answer>
                        <answer correct="true">2</answer>
                        <answer correct="false">3</answer>
                        <answer correct="false">4</answer>
                  </question>
                  <question>
                     ...
                  </question>
                     ...
            </content>
      </frame>
        ...
</alc>
```

Slika 19. ALC datoteka s pitanjima i odgovorima kao dodatnim sadržajem

Struktura ALC datoteke koja sadrži pitanja i odgovore kao dodatni sadržaj prikazana je na slici 19. Vrijednost atributa *type* je *quiz. Content* element sadrži jedan ili više *question* elemenata. *Question* element sadrži elemente: *questionText* i proizvoljan broj *answer* elemenata. *QuestionText* sadrži pitanje, a *answer* elementi sadrže ponuđene odgovore. Atribut *correct* označava koji je od ponuđenih odgovora točan. Jedno pitanje može imati više točnih odgovora.

4.1.4. Učitavanje ulaznih datoteka koristeći URL adrese datoteka

Ulazne datoteke se u LeCTo Player mogu učitati na dva načina: navodeći URL adresu ulazne datoteke i učitavanje datoteke s lokalnog diska koristeći formu za učitavanje datoteka (eng. *upload form*). Korisnik može kombinirati metode učitavanja ulaznih datoteka, na primjer: navodeći URL učitana je PSU datoteka i jedna ALC datoteka, a potom je s lokalnog diska učitano još nekoliko ALC datoteka.



Slika 20. Pozivi metoda prilikom učitavanja ulaznih datoteka

Forma za učitavanje ulaznih datoteka koristeći URL adrese datoteka implementirana je u *LeCTo.html* datoteci. Forma se sastoji od dva polja za unos teksta u koje se upisuje URL adresa ulazne datoteke. Jedna forma za unos teksta je za unos URL adrese PSU datoteke, a druga za unos URL adrese ALC datoteke.

```
<div id="webUpload">
<!-- Reading files from web dialog -->
PSU file URI:
<input id="webUriText" />
	<button class="webUpload" onclick="javascript:addFileUri();">
		Add PSU file</button>
<br />
<br />
	If you only want to upload ALC file, leave PSU file URI
	field empty.
<br />
ALC file URI:
<input id="webAlcUri" />
	<button class="webUpload" onclick="javascript:addAlcFileUri();">
		Add ALC file</button>
</div>
```

Kod 1. Implementacija forme za učitavanje datoteka navodeći URL

Na slici 20 prikazan je HTML kod kojim je implementirana forma za učitavanje ulaznih datoteka navodeći njihovu URL adresu. Kada korisnik pritisne gumb za učitavanje navedenih lokacija poziva se jedna od *javascript* metoda: `addFileUri()` ili `addAlcFileUri()`. Metode se nalaze u datoteci *LeCToPlayer.js*. Metode sadrže kod za čitanje navedenih adresa datoteka iz HTML forme te poziv metoda za prijenos datoteka (eng. *proxy*) s navedenih lokacija na LeCTo web poslužitelj.

```
function addFileUri() {

	var path = $("#webUriText").val();

	$.get("proxy.php", { 'url': path }, function(data) {
			updatePlayerFromWebUri(data);
		});
}
function addAlcFileUri() {
	var path = $("#webAlcUri").val();

	$.get("proxyOneUri.php", { 'url': path }, function(data) {
			updatePlayerFromWebUri(data);
		});
}
```

Kod 2. Metode za dohvat URL lokacije datoteka s HTML forme

Upisana URL adresa ulazne datoteke ne mora pokazivati na datoteku koja se nalazi u istoj domeni$_6$ kao i LeCTo Player. Zbog toga što se datoteke dohvaćaju koristeći jezik *javascript*, datoteke koje se ne nalaze u istoj domeni kao i LeCTo Player ne mogu biti dohvaćene. Ovo ograničenje postavlja sam *javascript* jezik zbog očuvanja pravila *same origin policy*. Pravilo *same origin policy* je sigurnosno pravilo koje nalaže da Internet preglednici ne smiju pristupati datotekama koje se nalaze na web poslužiteljima koji su u drugoj domeni$_6$ od Internet aplikacije koja je pokrenuta [8]. Zbog ovog ograničenja iz *javascript* metode se datoteci ne može izravno pristupiti putem navedene URL adrese. Zbog toga se URL adresa prosljeđuje *proxy.php* skripti. *Proxy.php* skripta se izvodi na LeCTo web poslužitelju[6]. Pošto se pravilo *same origin policy* odnosi samo na Internet pretraživače, ali ne i na poslužitelje, poslužitelj

[6] LeCTo web poslužitelj nije poseban web poslužitelj već web poslužitelj na kojem se nalazi LeCTo Player aplikacija

može dohvatiti sve datoteke, pa tako i one koje se ne nalaze u istoj domeni. *Proxy.php* skripta dohvaća PSU datoteku koja se nalazi na određenoj URL lokaciji. Skripta pročita sadržaj dohvaćene PSU datoteke kako bi se saznale adrese ostalih datoteka navedenih u PSU datoteci (CCE i ALC datoteke). Nakon što su sve adrese poznate skripta kopira datoteke na LeCTo web poslužitelj te vrati relativne putove (eng. *path*) do datoteka kako bi se mogle dohvatiti koristeći AJAX poziv iz *javascript* metode. Skripta *proxyOneUri.php* se poziva kada treba učitati ALC datoteku. U tom slučaju ne treba čitati sadržaj datoteke već se datoteka samo kopira na vlastiti poslužitelj.

```
$.get("proxy.php", { 'url': path }, function(data) {
      updatePlayerFromWebUri(data);
});
```

Navedenom *jquery* naredbom se poziva PHP skripta i prenosi URL parametar. Parametar `data` je povratna informacija skripte. U parametru `data` se nalaze relativni putovi do datoteka na LeCTo web poslužitelju.

Kada se neka ulazna datoteka učita, LeCTo Player prikaže korisniku listu učitanih datoteka. Metoda `updatePlayerFromWebUri(data)` služi za osvježavanje liste dohvaćenih datoteka. Iz ove metode se poziva metoda `getUploadedFile(file)` kojom se u dodaje datoteka u listu datoteka koje trebaju biti parsirane. Iz ove metode se poziva metoda `checkUploadedFiles()` koja provjerava koje su sve datoteke učitane. Ako su datoteke PSU i CCE učitane, LeCTo Player omogućuje reproduciranje snimke. Sve dok PSU i CCE datoteke nisu učitane, gumb koji pokreće snimku biti će onemogućen.

4.1.5. Učitavanje datoteka s lokalnog diska

Za unos datoteka s lokalnog diska koristi se forma za učitavanje datoteka (eng. *upload form*). Implementacija forme za učitavanje datoteka je gotova komponenta, *uploadify*[5]. *Uploadify* je nadogradnja (eng. *plugin*) za *jquery* koja omogućuje učitavanje više datoteka odjednom. Dodatak *uploadify* omogućuje jednostavnu ugradnju i vrlo je prilagodljiv. Da bi se dodatak integrirao s HTML stranicom, u zaglavlje stranice treba upisati sljedeće poveznice:

```
<script type="text/javascript" src="js/uploadify/swfobject.js"></script>
<script type="text/javascript"
src="js/uploadify/jquery.uploadify.v2.1.0.min.js"></script>
<script type="text/javascript" src="js/jquery.application.js"></script>
```

```
<link rel="stylesheet" type="text/css" href="css/uploadStyle.css"
	media="screen" />
```

Prikazane poveznice povezuju *LeCTo.html* stranicu s *uploadify* dodatkom. Logika *uploadify* forme za unos podataka podešava se u *jquery.application.js*

```
jQuery(document).ready(function() {

$('#mainftp').uploadify({
	'uploader'  : 'js/uploadify/uploadify.swf',
	'script'    : 'js/uploadify/uploadify.php',
	'multi'     : true,
	'auto'      : true,
	'height'    :'29',        //height of your browse button file
	'width'     :'150',       //width of your browse button file
	'sizeLimit' :'51200',     //remove this to set no limit on upload size
	'simUploadLimit' :'3',
	'buttonImg' : 'img/browseLocal.png',
	'cancelImg' : 'img/cancel.png',
	'folder'    : 'files/',          //folder to save uploads to

	onProgress: function() {
		$('#loader').show();
	},
	onAllComplete: function() {
		$('#loader').hide();
	},
	onComplete: function(event, queueID, fileObj, reponse, data) {
			$('#filesList').append("<li>"+fileObj.name+"</li>");

			getUploadedFile(fileObj.name);
	}
	});
	$('ul li:odd').addClass('odd');
});
```

Kod 3. Prikaz jquery.application.js datoteke

Kod 3 prikazuje *jquery.application.js* datoteku koja je konfiguracijska datoteka *uploadify* dodatka. U tablici 1 opisan je svaki od navedenih parametara u konfiguracijskoj datoteci.

Parametar	**Opis**
'uploader'	Aplikacija koja služi kao forma za učitavanje datoteka
'script'	PHP skripta koja učitava datoteke na web poslužitelj

'multi'	Je li omogućeno učitavanje više datoteka odjednom
'auto'	Treba li datoteke automatski učitavati
'height'	Visina gumba za pretragu datoteka (eng. *browse button*)
'width'	Širina gumba za pretragu datoteka
'sizeLimit'	Ograničenje veličine datoteka koje se mogu učitati
'simUploadLimit'	Ograničenje na broj datoteka koje se mogu istodobno učitavati
'buttonImg'	Slika gumba za pretragu datoteka
'cancelImg'	Slika gumba za prestanak učitavanja datoteke
'folder'	Datoteka na poslužitelju u koju će datoteke biti učitane
onProgress	Metoda kojom se određuje što će se obavljati dok je učitavanje u tijeku
onAllComplete	Metoda kojom se određuju radnje nakon što su sve datoteke učitane
onComplete	Metoda kojom se određuju radnje nakon što je jedna datoteka učitana

Tablica 1. Opis parametara *uploadify* forme za učitavanje datoteka

Uz navedene parametre postoje i drugi parametri i metode koje se nisu koristile u implementaciji alata LeCTo Player, a dostupne su i opisane u službenoj dokumentaciji dodatka [9]. U metodi `onAllComplete` se osvježava ispis učitanih datoteka za korisnika te se poziva metoda `getUploadedFile(fileObj.name)` kojom se učitana datoteka dodaje u listu za parsiranje. Iz metode `getUploadedFile` se poziva metoda `checkUploadedFiles()` kojom se provjeravaju učitane datoteke.

```
<div id="sidebar">
<!-- form to be replaced by uploadify -->
<form id="mainftp" action="upload.php" method="post"
enctype="multipart/form-data">
             <p><input type="file" name="file" id="file" /></p>
             <p><input type="submit" name="action" value="Upload" /></p>
      </form>
</div>
```

Kod 4. HTML kod za integraciju uploadify forme u LeCTo.html datoteci

4.1.6. Podatkovni model

Parsiranjem XML datoteka čitaju se podaci potrebni za konfiguraciju i pokretanje alata LeCTo Player. Kako bi se pročitani podaci mogli koristiti, trebaju biti pohranjeni u memoriju klijentskog računala. Posebnim funkcijama jezika *javascript* izgrađen je podatkovni model za pohranu podataka.

```
function PresentationFrame() {
        this.title = null;
this.startTime = null;
        this.endTime = null;
        this.contextID = null;
}
```

Kod 5. *Javascript* funkcija `PresentationFrame`

Funkcija prikazana isječkom koda 5 predstavlja *javascript* razred `PresentationFrame`. Programski jezik *javascript* ne podržava standardno kreiranje razreda kao u ostalim objektno orijentiranim programskim jezicima. Pisanjem metoda kao što je prikazano kodom 5 izgrađuje se *javascript* razred. Naredbom `new PresentationFrame()` instancira se objekt razreda.

Razred	Opis
`PresentationFrame`	Razred sadrži informacije o izmjeni sadržaja prezentacije (promjena slajda prezentacije, PDF stranice)
`ContentFrame`	Sadrži informacije o dodatnom sadržaju (iz ALC datoteke) za jedan vremenski period
`AdditionalLectureContent`	Sadrži informacije o dodatnom sadržaju (URL ALC datoteke). Sadržaj se čita iz PSU datoteke
`LectureVideo`	Sadrži informacije o video snimki prezentacije (iz PSU datoteke)
`NavigationWindow`	Sadrži informacije o navigacijskom dijalogu (iz PSU datoteke)
`UserNote`	Razred koji predstavlja korisničku bilješku za jedan vremenski period
`Question`	Razred koji predstavlja jedno pitanje (iz ALC datoteke)
`Answer`	Razred koji predstavlja jedan odgovor (iz ALC datoteke)
`Link`	Razred koji predstavlja jednu podatkovnu poveznicu (iz ALC datoteke)

Tablica 2. Podatkovni model alata LeCTo Player

4.1.7. Parsiranje XML datoteka

Nakon što su sve potrebne datoteke pohranjene na LeCTo poslužitelj trebaju biti prebačene na klijentsko računalo kako bi se mogle parsirati koristeći *javascript* programski jezik. Parsiranje počinje kada korisnik pritisne gumb „*Start Player*“. Pritiskom na gumb poziva se *javascript* metoda `getReadyForParsing()`:

```
<button id="startPlayer" onclick="getReadyForParsing();">Start Player</button>
```

Metoda `getReadyForParsing()` služi za sortiranje liste datoteka za parsiranje. Lista se sortira tako da prva na redu bude PSU datoteka, druga CCE datoteka, a tek onda slijede ALC datoteke. Ovakav redoslijed parsiranja je potreban jer se podaci pročitani iz PSU datoteke koriste prilikom parsiranja CCE i ALC datoteka. Iz metode `getReadyForParsing()` se poziva metoda `loadXML` koja kao parametar sadrži listu datoteka za parsiranje.

Zbog toga što je parsiranje implementirano koristeći *javascript* koji se izvodi na klijentskom računalu, datoteke moraju biti prebačene s poslužitelja u memoriju klijentskog računala. Prebacivanje datoteka se obavlja koristeći AJAX (*Asynchronous JavaScript and XML*) pozive[7]. U metodi `loadXML` stvara se objekt *XML DOM* koji predstavlja XML dokument pogodan za parsiranje[8]. Internet preglednici *Internet Explorer* i *Mozilla Firefox* na različit način instanciraju *XML DOM* objekt. Koristeći AJAX poziv u *XML DOM* objekt se pohranjuje sadržaj XML datoteke. Nakon što je sadržaj XML datoteke učitan, može se parsirati pozivom metode `readXML`.

```
if( window.ActiveXObject && /Win/.test(navigator.userAgent) ) {
xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
xmlDoc.async = false;
// parse PSU file first
xmlDoc.load("./files/" + xmlFiles[0]);
             readXML();
             ...
```

Kod 6. Kreiranje XML DOM objekta za Internet Explorer preglednik – metoda *loadXML*

```
else {
xmlDoc = document.implementation.createDocument("","",null);
        xmlDoc.async=false;
        // parse PSU file first
        xmlDoc.load("./files/" + xmlFiles[0]);
        readXML();
        ...
```

Kod 7. Kreiranje XML DOM objekta za Mozilla Firefox preglednik – metoda loadXML

Ako je uvjet:

```
if(window.ActiveXObject && /Win/.test(navigator.userAgent)) { ... }
```

zadovoljen radi se o *Internet Explorer* pregledniku.

Metoda `readXML` provjerava o kojoj se ulaznoj datoteci radi: PSU, CCE ili ALC. Provjera se obavlja tako da se pročita korijenski element XML datoteke. Kada je određen tip ulazne datoteke, poziva se jedna od ogovarajućih metoda: `parsePSU`, `parseCCE` ili `parseALC`.

```
Function readXML() {
         if(xmlDoc.documentElement.tagName == "psu") {
               // we are parsing PSU file
               parsePSU();
        }
        else if(xmlDoc.documentElement.tagName == "alc") {
               // we are parsing ALC file
               parseALC();
        }
        else if(xmlDoc.documentElement.tagName == "presentation") {
               // we are parsing CCE file
               parseCCE();
        }
}
```

Kod 8. Prikaz metode `readXML`

Parsiranje XML datoteka se obavlja koristeći metode objekta *XML DOM*[9]. Objekt nudi metode za dohvaćanje elementa i atributa iz XML strukture.

```
// getting the video URI from <uri> element
var uriElement = videoElements[i].getElementsByTagName("uri");
if(uriElement[0].childNodes[0] != null) {
videoObject.uri = uriElement[0].childNodes[0].nodeValue;
}
```

Kod 9. Primjer dohvaćanja vrijednosti elementa *XML DOM* objekta

Nakon što su sve učitane XML datoteke parsirane, a njihov sadržaj pohranjen u memoriju, poziva se metoda `customizeThePlayer`. Metoda `customizeThePlayer` pokreće postupak inicijalizacije i prikaza LeCTo Player korisničkog sučelja.

4.2. Inicijalizacija i prikaz LeCTo Player sučelja

4.2.1. Jquery *User Interface*

Za izradu korisničkog sučelja korišten je dodatak programske knjižnice (eng. *library*) *jquery*, *jquery user interface* (*jquery UI*)[10]. Dodatak *iquery UI* je skup komponenti za implementaciju grafičkog sučelja poput: gumba, dijaloga (eng. *dialog*), izbornika datuma i slično. Komponente se lako integriraju u *javascript* kod zbog toga što se svakoj komponenti može pristupiti korištenjem *jquery* naredbi. Svaka komponenta ima skup dostupnih: svojstava (eng. *options*), metoda (eng. *methods*) i događaja (eng. *events*) [10]. Korištenjem svojstava, metoda i događaja komponente su potpuno prilagodljive raznim zahtjevima.

Dodatak *jquery UI* se integrira s HTML stranicom dodavanjem sljedećih linija HTML koda u zaglavlja HTML stranice:

```
<link type="text/css" href="css/ui-lightness/jquery-ui-1.8.11.custom.css"
rel="Stylesheet" />
<script type="text/javascript" src="js/jquery-ui-1.8.11.custom.min.js"></script>
```

```
<!-- kod unutar HTML stranice -->
<div id="permutedIndex">
<!-- sadržaj ovog div elementa bit će unutar novog prozora  -->
</div>
<script>
	$('#permutedIndex').dialog('open');
	$('#permutedIndex').dialog( "option", "title", 'Naslov prozora' );
</script>
```

Kod 10. Primjer kreiranja novog dijaloga koristeći *jquery UI*

4.2.2. Inicijalizacija LeCTo Player sučelja

Nakon što su XML datoteke parsirane, a njihov sadržaj pohranjen u memoriju, poziva se metoda `customizeThePlayer` koja pokreće postupak inicijalizacije korisničkog sučelja. Metoda se sastoji od niza *jquery* i *javascript* naredbi kojima se inicijaliziraju *jquery UI* komponente.

```
// jump to player tab
	$( "#tabs" ).tabs("select" , "#player");
	// enable player controls
	$( "button", "#player").button({ disabled: false });

	$("#playerDiv1").dialog("open");
	$("#playerDiv2").dialog("open");
	$("#navIndex").dialog("open");
```

Kod 11. Početak metode `customizeThePlayer`

```
for(var i=0; i<lectureVideos.length; i++) {
// load the video file into the player
jwplayer('player'+ (i+1)).load({ title: 'video', file: lectureVideos[i].uri });

// resize the player
jwplayer('player'+ (i+1)).resize(lectureVideos[i].windowWidth,
lectureVideos[i].windowHeight);

$("#playerDiv" + (i+1) ).dialog( "option", "width", width);
$("#playerDiv" + (i+1) ).dialog( "option", "height", height);
// set the window title
$("#playerDiv" + (i+1)).dialog("option","title",lectureVideos[i].windowTitle);
// set resizable option
if(lectureVideos[i].isWindowResizable == "true")
$("#playerDiv" + (i+1)).dialog( "option", "resizable", true);
else
       $("#playerDiv" + (i+1)).dialog( "option", "resizable", false);

// set the position of window
$("#playerDiv" + (i+1)).dialog( "option", "position",
[parseInt(lectureVideos[i].windowPositionX),
       parseInt(lectureVideos[i].windowPositionY)]);
}
```

Kod 12. Konfiguriranje video dijaloga

Kod 11 prikazuje početak metode `customizeThePlayer`. Navedenim naredbama se omogućavaju (eng. *enable*) kontrole te se otvaraju tri osnovna dijaloga alata LeCTo Player: prvi video dijalog, drugi video dijalog i navigacijski dijalog. Navedene naredbe slijedi *for* petlja kojom se video dijalozi konfiguriraju kako je navedeno u PSU datoteci (element *window*). Konfiguriranje video dijaloga prikazano je isječkom koda 12.

```
jwplayer('player'+ (i+1)).load({ title: 'video', file: lectureVideos[i].uri});
jwplayer('player'+ (i+1)).resize(lectureVideos[i].windowWidth,
lectureVideos[i].windowHeight);
```

Izdvojene naredbe su naredbe API sučelja komponente *jwplayer*. Komponenta *jwplayer* opisana je u poglavlju „Sinkronizacija i prikaz video snimki“. Navedenim naredbama u

komponentu se učitava video snimka te se podešava veličina komponente *jwplayer* tako da odgovara veličini video dijaloga.

```
// creating the list of context titles for navigation
var indexDiv = document.getElementById("navIndex");
indexDiv.innerHTML = "";

for(var i=0; i<presentationTimeline.length; i++) {
       indexDiv.innerHTML += "<a href='javascript:seekPlayerTo(" +
                        presentationTimeline[i].startTime + ")'>" +
                        presentationTimeline[i].title + "</a><br />";
}
// customizing content navigation window
$("#navIndex").dialog("option", "title", navigationWindow.windowTitle);
$("#navIndex").dialog("option", "width",
parseFloat(navigationWindow.windowWidth));
$("#navIndex").dialog("option", "height",
parseFloat(navigationWindow.windowHeight));
$("#navIndex").dialog( "option", "position",
[parseInt(navigationWindow.windowPositionX),
 parseInt(navigationWindow.windowPositionY)]);
```

Kod 13. Kreiranje navigacijskog dijaloga

Nakon što su video dijalozi konfigurirani, metoda stvara navigacijski dijalog. Navigacijski dijalog sadrži nazive svih većih cjelina predavanja, najčešće su to nazivi slajdova *PowerPoint* prezentacije. Svaki naziv je ujedno i poveznica koja, kada je pritisnuta, prebacuje snimku na određeni dio.

Najveći dio `customizeThePlayer` metode je stvaranje dijaloga za prikaz dodatnog sadržaja. Stvaranje dijaloga za prikaz dodatnog sadržaja počinje u *for* petlji. *For* petlja se obavlja onoliko puta koliko ima različitog dodatnog sadržaja:

```
for(var i=0; i<additionalLectureContentIds.length; i++)
```

Lista `additionalLectureContentIds` sadrži oznake svih različitih dodatnih sadržaja (onoliko koliko je ALC datoteka učitano). Za svaki različiti dodatni sadržaj stvorit će se posebni dijalog u kojemu će biti prikazivan taj sadržaj. Da bi se mogao stvoriti dijalog pomoću dodatka *jquery UI* u HTML stranici mora postojati *div* element koji će predstavljati taj dijalog. Kako se ne

može unaprijed znati koliko će različitih dodatnih sadržaja snimka sadržavati (a samim tim i različitih dijaloga), *div* elementi moraju se stvoriti dinamički.

```
var divClassWithSpaces = additionalLectureContentIds[i];
       var divClass = clearSpaces(divClassWithSpaces);

var newDiv = document.createElement("div");
       newDiv.id = divClass;
       document.body.appendChild(newDiv);

       divClass = "#" + divClass;
```

Kod 14. Isječak koda za dinamičko kreiranje *div* elemenata HTML stranice

Isječak koda 14 prikazuje kod za dinamičko generiranje *div* elemenata u HTML stranici. Naredbama

```
var newDiv = document.createElement("div");
newDiv.id = divClass;
```

stvara se novi *div* element te mu se zadaje naziv atributa *id.*

```
// opening new dialog for additional content
$(function() {
      $( divClass ).dialog({ title: additionalLectureContentIds[i] },
          { beforeClose: function(event, ui) {
              // before this dialog closes we must
              // make it possible for a user to reopen it again
              // without restarting a lecture
              var reopenDiv = document.getElementById("contentDialogs");
              var title = $( this ).dialog( "option", "title" );
              if(reopenDiv.innerHTML.indexOf(title) == -1) {
                     var dialogId = clearSpaces(title);
                     reopenDiv.innerHTML +=
"<li><a href='javascript:reopenDialog(\""+dialogId+"\")'>"
                            + title +"</a></li>";
              }
       }
});
```

Kod 15. Kod za dinamičko kreiranje dijaloga za prikazivanje dodatnog sadržaja

Nakon što je potrebni *div* element, može se stvoriti i potrebni dijalog. Funkcija koja stvara dijalog prikazana je isječkom koda 15. Nakon što je dijalog stvoren, automatski se prikazuje korisniku.

Zadnje što preostaje kod inicijalizacije sučelja je konfiguriranje parametara dijaloga tako da odgovaraju parametrima navedenim u PSU ili ALC datoteci (*window* element). U isječku koda 15 prikazan je kod za konfiguriranje parametara prozora.

```
$(divClass).dialog( "option", "title", alcTitle );
$(divClass).dialog( "option", "width", alcWidth);
$(divClass).dialog( "option", "height", alcHeight);
$(divClass).dialog( "option", "resizable", alcIsResizable);
$(divClass).dialog( "option", "position",
 [parseInt(alcPosX),parseInt(alcPosY)]);
```

Kod 16. Dio koda za konfiguriranje parametara dijaloga za prikazivanje dodatnog sadržaja

Na kraju metode `customizeThePlayer` poziva se metoda `startRefreshing` koja inicijalizira komponentu za automatsko osvježavanje dodatnog sadržaja.

4.3. Sinkronizacija i prikaz video snimki

Da bi se video snimke mogle reproducirati na web stranici, potrebno je u web stranicu ugraditi (eng. *embed*) alat za reproduciranje multimedijalnih snimki (eng. *media player*). Za implementaciju alata LeCTo Player korištena je gotova komponenta za reproduciranje multimedijalnih komponenti, *jwplayer*[4]. Zbog toga što LeCTo Player reproducira dvije video snimke koje se u svakom trenutku moraju izvoditi paralelno, u web stranicu *LeCTo.html* ugrađena su dvije komponente *jwplayer* koje su posebnim metodama sinkronizirane.

4.3.1. *Jwplayer* ugradbena komponenta za reproduciranje video snimki

Jwplayer je ugradbena komponenta za reproduciranje video snimki (eng. *embedded video player*) formata *flash video* (FLV). Komponenta je izrađena koristeći tehnologiju *flash.* Zbog korištenja tehnologije *flash,* komponenta se može ugraditi u sve poznatije Internet preglednike, a može se izvoditi na bilo kojem operacijskom sustavu. Ugradbena komponenta *Jwplayer* je odabrana zbog: mogućnosti lake integracije u web stranicu, postojanje API-a za

integraciju s *javascript* jezikom, mogućnost rada s HTML 5 standardom (buduća unaprjeđenja alata LeCTo) i mogućnosti personalizacije izgleda. API komponente *jwplayer* pruža skup metoda, svojstava i događaja koji omogućuje prilagodljiv rad s komponentom.

Prije prikaza načina ugradnje i implementaciju sinkronizacijskih metoda, potrebno je razlikovati načine na koji se datoteka sa snimkom (FLV datoteka) prenosi s udaljenog poslužitelja na klijentsko računalo.

4.3.2. Prijenos snimke s udaljenog poslužitelja na klijentsko računalo

Datoteke s video snimkama se ne moraju nalaziti na poslužitelju iz iste domene kao i alat LeCTo Player. Komponenta *jwplayer* podržava dva načina prijenosa video snimki: HTTP protokolom i $RTMP_1$ protokolom.

Prijenos koristeći HTTP protokol

HTTP protokol se koristi za prijenos kada je video snimka pohranjena na poslužitelj koji ne podržava RTMP protokol. Kada se koristi HTTP protokol, korisnik ne može prebaciti i gledati onaj dio snimke koji još nije učitan.

Slika 21. Ilustracija nedostatka prijenosa snimke HTTP protokolom

Slika 21 prikazuje nedostatak prijenosa video snimke koristeći HTTP protokol. Na slici 21 plava strjelica prikazuje koliko je video snimke prebačeno, tj. učitano na klijentskom računalu. Ako korisnik želi prebaciti snimku na mjesto označeno zelenom strjelicom, snimka će se uspješno prebaciti na odabrano mjesto i nastaviti reproducirati. Ako korisnik želi prebaciti snimku na mjesto označeno crvenom strjelicom, snimka se neće prebaciti jer taj dio snimke još nije učitan. Korisnik će moći prebaciti snimku na mjesto gdje pokazuje crvena strjelica tek kada se taj dio snimke učita, tj. prenese s udaljenog poslužitelja na klijentsko računalo.

Prijenos koristeći RTMP protokol

Ako udaljeni poslužitelj podržava RTMP protokol, snimka će se prebacivati koristeći RTMP protokol (eng. *Real Time Messaging Protocol*). Koristeći RTMP protokol rješava se problem prijenosa video snimke HTTP protokolom.

Slika 22. Ilustracija rada RTMP protokola

Na slici 22 ilustriran je rad RTMP protokola. Za razliku od prijenosa HTTP protokolom, RTMP protokol omogućuje korisniku prebacivanje snimke i na one dijelove koji još nisu učitani. Ako korisnik prebaci snimku na dio koji još nije učitan, udaljeni poslužitelj će početi slati dijelove snimke koje korisnik zahtjeva pa će korisnik moći gledati dio snimke koji je odabrao.

4.3.3. Integracija komponente s alatom LeCTo Player

Prvi korak u integraciji komponente je povezivanje potrebnih datoteka s HTML stranicom. Upisivanjem sljedećih HTML linija koda povezuju se *jwplayer* datoteke s HTML stranicom:

```
<script type="text/javascript" src="js/jquery-1.5.1.min.js"></script>
<script type="text/javascript" src="jwplayer/jwplayer.js"></script>
```

Datoteka `jquery-1.5.1.min.js` je datoteka koja sadrži programsku knjižnicu *jquery*, a potrebna je jer API komponente *jwplayer* koristi *jquery* naredbe. Datoteka `jwplayer.js` je implementacija komponente *jwplayer*. Sve potrebne datoteke mogu se besplatno preuzeti sa službene stranice komponente *jwplayer*. Sljedeći korak u integraciji je ugradnja (eng. *embed*) komponente *jwplayer* u web stranicu. Koriste se različite metode ugradnje za *Internet Explorer* preglednik i *Mozilla Firefox* preglednik.

```
<!--[if IE]>
<div id="playerDiv1">
<object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
width='300' height='300' id='player1' name='player1'>
<param name='movie' value='jwplayer/player.swf'>
<param name='allowfullscreen' value='true'>
<param name='allowscriptaccess' value='always'>
<param name='flashvars' value='stretching=exactfit'>
</object>
</div>
<div id="playerDiv2">
<object classid='clsid:D27CDB6E-AE6D-11cf-96B8-444553540000'
width='300' height='300' id='player2' name='player2'>
       <param name='movie' value='jwplayer/player.swf'>
       <param name='allowfullscreen' value='true'>
       <param name='allowscriptaccess' value='always'>
       <param name='flashvars'
value='skin=./jwplayer/skins/five.zip&stretching=exactfit'>
</object>
</div>
<![endif]-->
```

Kod 17. Kod za ugradnju komponente *jwplayer* u *Internet Explorer* preglednik

```
<![if !IE]>
<div id="playerDiv1">
<embed id='player1'
             name='player1'
             src='jwplayer/player.swf'
             width='300'
             height='300'
             allowscriptaccess='always'
             allowfullscreen='true'
             flashvars="file="
/>
</div>
```

```
<div id="playerDiv2">
	<embed id='player2'
		name='player2'
		src='jwplayer/player.swf'
		width='300'
		height='300'
		allowscriptaccess='always'
		allowfullscreen='true'
		flashvars='skin=./jwplayer/skins/five.zip'
	/>
</div>
<![endif]>
```

Kod 18. Kod za ugradnju komponente *jwplayer* u *Mozilla Firefox* preglednik

Kod za ugradnju komponente sadrži razne parametre kojima se konfigurira komponenta. Opis korištenih parametara prikazan je tablicom 2.

Parametar	**Opis**
name	Naziv komponente
`'movie'`/src	Put do datoteke s komponentom *jwplayer*
id	Jedinstvena oznaka komponente.
width	Širina komponente
height	Visina komponente
allowscriptaccess	Parametar kojim se određuje hoće li se komponenti moći pristupati koristeći *javascript* jezik.
flashvars	Parametri koji se odnose na *flash* komponentu. LeCTo Player koristi `file` i `skin` varijable. `File` varijabla označava video snimku koju komponenta treba pokrenuti. `Skin` varijabla određuje izgled komponente.

Tablica 3. Parametri komponente *jwplayer*

4.3.5. Sinkronizacija komponenti *jwplayer*

LeCTo Player omogućuje reproduciranje dvije paralelne video snimke. Svaku snimku reproducira posebna komponenta *jwplayer*. Da bi snimke bile sinkronizirane, potrebne su metode za sinkronizaciju te dvije komponente. Metode za sinkronizaciju nalaze se u *LeCTo.html* datoteci.

```
<script type='text/javascript'>

jwplayer('player2').onReady(function(event) {});
// Cathing events of the players
 jwplayer('player2').onBuffer(function(event) {
                    jwplayer('player1').play(true);
              });
jwplayer('player2').onPlay(function(event) {
                    jwplayer('player1').play(true);
              });
jwplayer('player2').onPause(function(event) {
                    jwplayer('player1').pause(true);
              });
jwplayer('player1').onBuffer(function(event) {
                    jwplayer('player2').play(true);
              });
jwplayer('player1').onPlay(function(event) {
              jwplayer('player2').play(true);
              });
jwplayer('player1').onPause(function(event) {
              jwplayer('player2').pause(true);
       });
jwplayer('player1').onSeek(function(event) {
              jwplayer('player2').seek(event.offset);
});

</script>
```

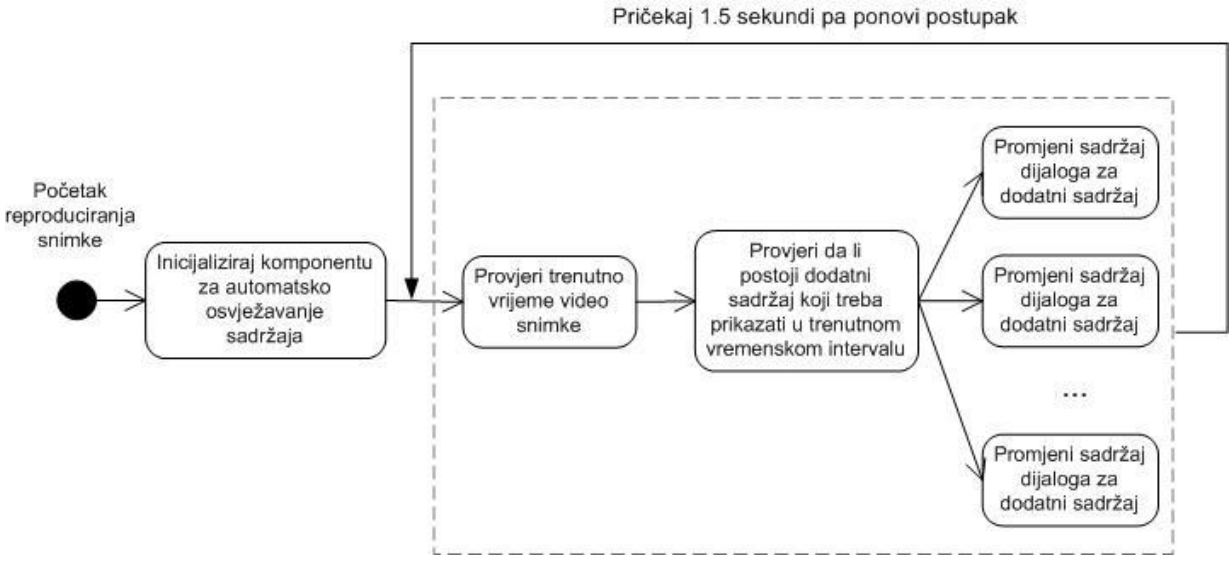
Kod 19. Metode za sinkronizaciju komponenti *jwplayer*

Osnova sinkronizacije je slušanje određenih događaja komponenti *jwplayer*. Događaj `onBuffer` se okida (eng. *trigger*) kada se video snimka počne učitavati. Događaj `onPause` se okida kada se video snimka pauzira. Događaj `onPlay` se okida kada se video snimka pokrene. Događaj

`onSeek` se okida prilikom navigacije po video snimci. Kada se dogodi određeni događaj na jednoj od komponenti, isti događaj se inicira na drugoj komponenti, npr. kada se prva video snimka pauzira, pozvat će se metoda kojom će se pauzirati druga video snimka.

4.4. Sinkronizacija dodatnog sadržaja

Osnovna funkcionalnost alata LeCTo Player je obogaćivanje video snimki dodatnim sadržajem koji je sinkroniziran sa snimkom. Sadržaj se sinkronizira tako da autor dodatnog sadržaja odabire točne vremenske trenutke tijekom video snimke kada će određeni sadržaj biti prikazan gledateljima. Cilj sinkronizacije je prikazivanje točno određenog sadržaja u točno određenom vremenskom intervalu.



Slika 23. Operacije sinkronizacije dodatnog sadržaja s video snimkom

Na slici 23 prikazan je dijagram s aktivnostima koje se obavljaju za vrijeme sinkronizacije dodatnog sadržaja. Komponenta za automatsko osvježavanje sadržaja inicijalizira se metodom `startRefreshing` nakon postupka inicijalizacije LeCTo Player sučelja. Komponenta za automatsko osvježavanje sadržaja svakih 1500 milisekundi obavlja tri operacije: provjera trenutnog vremena video snimke, provjera postoji li dodatni sadržaj koji treba biti prikazan u trenutnom intervalu i osvježavanje svakog od potrebnih dijaloga za dodatni sadržaj.

```
// refresher - component for content refreshing
function startRefreshing() {
	refresher = setInterval("updateAdditionalLecuteContent()", 1500);
}

// Method that stops refresher
function stopRefreshing() {
	clearInterval(refresher);
}
```

Kod 20. Funkcije za incijalizaciju i zaustavljanje komponente za automatsko osvježavanje sadržaja

Javascript naredba:

```
setInterval("updateAdditionalLecuteContent()", 1500);
```

omogućuje automatsko izvođenje *javascript* koda, u ovom slučaju pozivat će se metoda `updateAdditionalLecuteContent` svakih 1500 milisekundi. Metoda će se pozivati sve dok se metodom `clearInterval` ne zaustavi.

Metoda `updateAdditionalLecuteContent` sadrži kod za: provjeru trenutnog vremena video snimke, provjeru postoji li dodatni sadržaj koji treba biti prikazan u trenutnom intervalu i osvježavanje svakog od potrebnih dijaloga za dodatni sadržaj.

```
function updateAdditionalLecuteContent() {
	// we need to find out current time of the lecture video
	var currentTime = jwplayer('player1').getPosition();
```

Kod 21. Kod za dohvaćanje trenutnog vremena video snimke

Isječkom koda 20 prikazan je početak metode `updateAdditionalLecuteContent`. Na početku metode se dohvaća trenutno vrijeme video snimke naredbom dostupnom u *jwplayer* API knjižnici:

```
var currentTime = jwplayer('player1').getPosition();
```

Dovoljno je da se trenutno vrijeme očita samo s jedne komponente *jwplayer* jer su obadvije komponente *jwplayer* međusobno sinkronizirane pa pokazuju isto vrijeme.

Sav dodatni sadržaj se nalazi unutar objekata `contentFrame`. Lista `contentTimeline` sadrži sve `contentFrame` objekte. Svaki `contentFrame` objekt predstavlja dodatni sadržaj koji treba biti prikazan u određenom vremenskom intervalu.

```
function ContentFrame() {
	this.type = null;
	this.windowId = null;
	this.startTime = null;
	this.endTime = null;
	this.duration = null;
	this.content = null;
	this.windowTitle = null;
	this.windowWidth = null;
	this.windowHeight = null;
	this.isWindowResizable = null;
	this.isWindowOnTop = null;
	this.isWindowAutoOn = null;
	this.windowPositionX = null;
	this.windowPositionY = null;
	this.questions = null;
	this.links = null;
}
```

Kod 22. `ContentFrame` *javascript* razred

Isječkom koda 22 prikazan je `ContentFrame` razred. Crveno označeni atributi se koriste prilikom osvježavanja dodatnog sadržaja. Objekti ovog razreda se instanciraju i njima se popunjava lista `contentTimeline` prilikom parsiranja ALC ulaznih datoteka.

Nakon što je poznato trenutno vrijeme snimke, u *for* petlji se prolazi kroz `contentTimeline` listu te se traže oni `ContentFrame` objekti koji zadovoljavaju uvjet:

```
contentTimeline[i].startTime <= currentTime &&
contentTimeline[i].endTime > currentTime
```

Ako je navedeni uvjet zadovoljen, sadržaj `content` atributa trenutnog `ContentFrame` objekta treba prikazati korisniku u točno određenom dijalogu. Dijalog se određuje na temelju `windowId` atributa istog `ContentFrame` objekta. Atribut `windowId` objekta `ContentFrame` i *id* atribut *div* elementa traženog dijaloga se podudaraju.

```
var wantedDialog = contentTimeline[i].windowId;
wantedDiv = clearSpaces(wantedDialog);
```

Navedenim linijama koda dohvaća se *div* element traženog dijaloga. Kada je poznat dijalog čiji sadržaj treba biti osvježen, *jquery* naredbom se mijenja sadržaj dijaloga. Sadržaj naredbi za izmjenu sadržaja dijaloga ovisi o tipu dodatnog sadržaja (tekst, podatkovne poveznice, pitanja i odgovori ili web stranica).

```
if(contentTimeline[i].type == "other") {
$("#" + wantedDiv).html(contentTimeline[i].content);
}
```

Kod 23. Izmjena sadržaja dijaloga za prikaz dodatnog sadržaja - tekstualni sadržaj

```
else if(contentTimeline[i].type == "web") {
	var pageToShow = contentTimeline[i].content;
// if there is no „http“ in web page link, add it
	if(contentTimeline[i].content.indexOf("http") == -1) {
		pageToShow = "http://" + pageToShow;
	}
// create HTML code for new iframe element
var iframeString = "<iframe src=\""+ pageToShow +"\""+
				"width='100%' height='100%'><iframe>";

	// update dialog window only if there is a new web page to show
	// if old page needs to be shown don't update the dialog because
// if dialog is updated web page will be refreshing every 1.5 seconds
	if(oldWebPage[contentTimeline[i].windowTitle] !=
contentTimeline[i].content) {
		$("#" + wantedDiv).html(iframeString);
oldWebPage[contentTimeline[i].windowTitle] =
contentTimeline[i].content;
	}
}
```

Kod 24. Izmjena sadržaja dijaloga za prikaz dodatnog sadržaja - web stranica

Naredba `$("#" + wantedDiv).html(...)` unutar *div* elementa s atributom *id* vrijednosti `wantedDiv` upisuje HTML kod `contentTimeline[i].content`.

```
else if(contentTimeline[i].type == "links") {
	// list that contains all the links for this frame
var links = contentTimeline[i].links;
var linksHtml = "";
	for(var k=0; k<links.length; k++) {
		// create HTML code for every link in the list
		linksHtml += "<a href=\"javascript:openLinkInNewWindow('" +
		links[k].linkTo + "')\">" + links[k].linkTitle + "</a><br />";
	}
	// add links HTML code to the div element of the dialog to be
	// shown in the dialog body
	$("#" + wantedDiv).html(linksHtml);
}
```

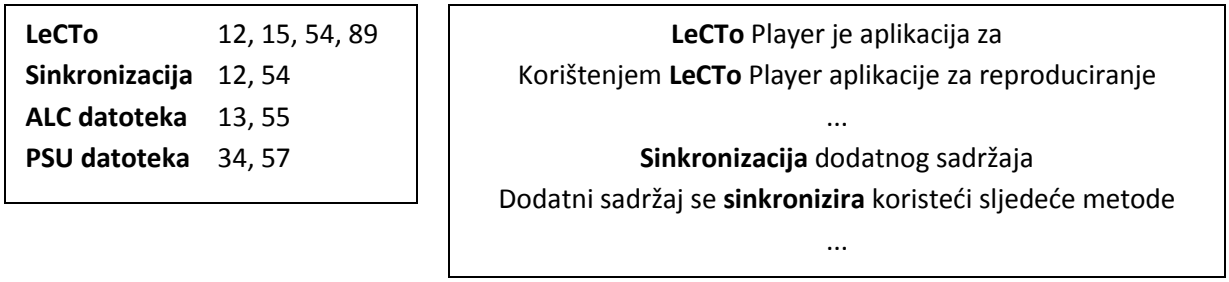
Kod 26. Izmjena sadržaja dijaloga za prikaz dodatnog sadržaja - podatkovne poveznice

```
else if(contentTimeline[i].type == "quiz") {
  var quizHtml = "";
  var questionsArray = contentTimeline[i].questions;
  for(var k=0; k < questionsArray.length; k++) {
	// create HTML code for question
	quizHtml += "<b>" + (k+1) + ". " + questionsArray[k] + "</b><br />";
	var answersArray = questionsArray[k].answersList;
	  // create HTML code for every answer
	  for(var n=0; n<answersArray.length; n++) {
		quizHtml += "<input type=\"radio\" name=\"group" + (n+1) +
			"\" onclick=\"javascript:checkAnswer('answer"+
			k+""+n+"',"+ answersArray[n].isCorrect +")\" value=\""
			answersArray[n].isCorrect + "\">" +
			"<span id=\"answer"+k + "" + n +"\">"+
			answersArray[n].answerText+ "</span><br />";
		if(n == (answersArray.length-1)) {
			quizHtml += "<br />"
		}
	  }
  }
  // show all the questions to the dialog
  $("#" + wantedDiv).html(quizHtml);
}
```

Kod 25. Izmjena sadržaja dijaloga za prikaz dodatnog sadržaja - pitanja i odgovori

4.5. Implementacija permutiranog indeksa

Permutirani indeks je specijalni oblik indeksa koji ključne riječi stavlja u kontekst u kojem se nalaze. Na slici 24 prikazana je razlika između standardnog indeksa i permutiranog indeksa.



Slika 24. Usporedba standardnog indeksa i permutiranog indeksa - permutirani indeks je desno

Uz navigaciju snimkom koristeći navigacijski indeks s nazivima osnovnih cjelina predavanja, LeCTo Player omogućuje i navigaciju snimkom koristeći permutirani indeks. Svaka ključna riječ permutiranog indeksa je poveznica na određeni dio snimke.

Zbog ograničene brzine izvođenja *javascript* jezika, permutirani indeks za veće količine teksta nije moguće implementirati koristeći *javascript* jezik. Za stvaranje permutiranog indeksa korišten je PHP skriptni jezik. Pritiskom na gumb „*Create index*“ u alata LeCTo Player poziva se *javascript* metoda `createPermutedIndex`. Unutar metode se dodatni sadržaj (atribut `content`) svakog `contentFrame` objekta iz `contentTimeline` liste prebaci u mapu `allContentMap`. Ključ svakog elementa mape je vrijeme kada određeni sadržaj treba prikazati, a vrijednost svakog elementa je dodatni sadržaj koji odgovara vremenu (ključu elementa). Nakon što je sav dodatni sadržaj prebačen u mapu, mapa se POST metodom šalje na LeCTo poslužitelj PHP skripti *permutedIndexGenerator.php.*

```
// send all content of the lecture to the web server
// where permuted index will be created
$.post("permutedIndexGenerator.php", { 'map': allContentMap },
function(data) {
$("#permutedIndex").html(data);
});
```

Kod 27. POST metoda za slanje mape `allContentMap` PHP skripti za kreiranje permutiranog indeksa

Svaka ključna riječ permutiranog indeksa je izgrađena uzimajući tri riječi iz desnog konteksta i tri riječi iz lijevog konteksta ključne riječi. Ključna riječ je svaka riječ dodatnog sadržaja koja je ima više od tri znaka.

```
if(strlen($contentArray[$i]) > 3) {
// creating left context
	for($j=3; $j>0; $j--) {
		if($i-$j >= 0) {
$tempIndex->leftContext .= $contentArray[$i-$j] . " ";
		}
}
// creating right context
	for($j=1; $j<=3; $j++) {
		if($i+$j < sizeof($contentArray)) {
			$tempIndex->rightContext .= " " . $contentArray[$i+$j];
		}
	}
```

Kod 28. Kreiranje lijevog i desnog konteksta ključne riječi - *permutedIndexGenerator.php*

Nakon što je za svaku ključnu riječ stvori redak permutiranog indeksa, PHP skripta vraća HTML kod permutiranog indeksa u obliku niza podatkovnih poveznica te lijevog i desnog konteksta. Svaka podatkovna poveznica predstavlja jednu ključnu riječ. Pritiskom na podatkovnu poveznicu poziva se metoda koja prebacuje snimku na dio gdje se govori o ključnoj riječi.

```
foreach($permutedIndex as $value) {
	$link = "<a href='javascript:seekPlayerTo(" . $value->time . ");'>" .
	"  <b>" . $value->keyword . "</b>  </a>";

	echo $value->leftContext . $link . $value->rightContext . "<br />";
}
```

Kod 29. Generiranje permuitranog indeksa - *permutedIndexGenerator.php*

4.6. Generiranje PSU datoteke

Kada LeCTo Player reproducira snimku predavanja obogaćenu s mnogo dodatnog sadržaja, otvoreno je mnogo dijaloga. Dva dijaloga sadrže video snimke, jedan dijalog je navigacijski, a za svaku učitanu ALC datoteku otvoren je posebni dijalog. LeCTo Player omogućuje korisnicima razmještanje, promjenu veličine i zatvaranje dijaloga. Kada se dijalozi poslože po ekranu, korisnik može stvoriti novu PSU datoteku koja će uz lokacije ulaznih datoteka sačuvati razmještaj i veličinu svih dijaloga. Na taj način korisnik ne mora ponovo razmještati dijaloge po ekranu već je dovoljno da učita stvorenu PSU datoteku. PSU datoteka se stvara pritiskom na gumb „*Create PSU file*“. Pritiskom na gumb pokreće se *javascript* metoda `createPsuFile` u datoteci *LeCToPlayer.js*.

Pošto je PSU datoteka XML datoteka, potrebno je izgraditi XML dokument s identičnim elementima učitane PSU datoteke, uz izmjenu konfiguracijskih parametara (*window* elementa). Izgradnja XML strukture obavlja se pomoću `XMLWriter` razreda. Implementacija `XMLWriter` razreda se nalazi u datoteci *XMLWriter.js*. Razred pruža sve potrebne metode za izgradnju XML strukture. Objekt `XMLWriter` razreda se instancira naredbom:

```
var XML = new XMLWriter( 'UTF-8', '1.0' );
```

```
XML.BeginNode("size");
XML.WriteString("\n");
       XML.Node("width", jwplayer('player'+(i+1)).getWidth().toString());
       XML.WriteString("\n");
       XML.Node("height", jwplayer('player'+(i+1)).getHeight().toString());
       XML.WriteString("\n");
XML.EndNode();
```

Kod 30. Isječak koda za generiranje XML strukture PSU datoteke – metoda `createPsuFile`

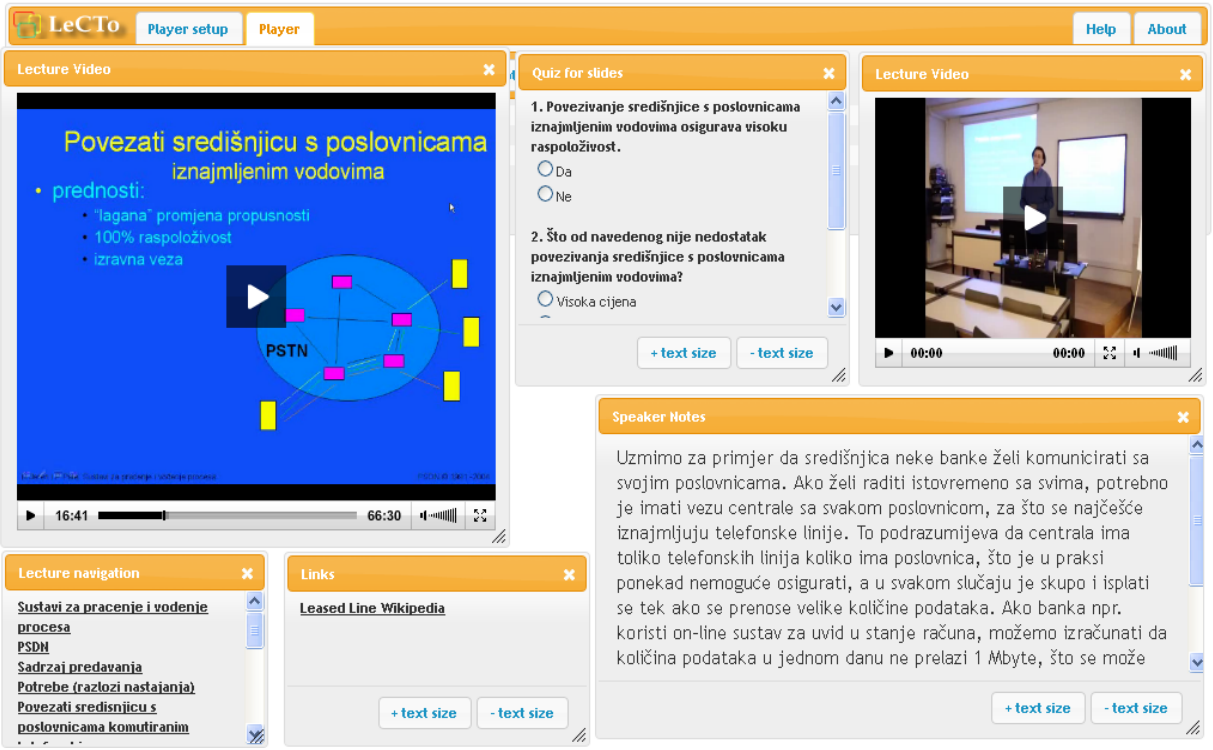
Nakon što je izgrađena struktura cijelog dokumenta, mora se stvoriti datoteka. Stvorena datoteka se mora pohraniti na klijentsko računalo. Zbog sigurnosnog pravila web preglednika, *javascript* jezik ne dopušta pisanje i spremanje datoteka na klijentsko računalo. Zbog toga se XML struktura u obliku *string* parametra šalje PHP skripti *psuGenerator.php*. Skripta *psuGenerator.php* koristeći ulazni parametar stvara XML datoteku te ju pohranjuje na poslužitelj, a klijentskom računalu vraća put do datoteke kako bi ju korisnik mogao preuzeti.

```php
<?php
	$srcFolder = "files/";
	$fileName = $srcFolder."lectureSetUp.psu.xml";

	// create new file
	$fileHandle = fopen($fileName, 'w') or die("can't open file");

	// get the xml string from POST variable
	$dataToWrite = $_POST["data"];

	if($fileHandle)
{

	if(!fwrite($fileHandle, $dataToWrite))
		die("couldn't write to file.");
}

	fclose($fileHandle);
	// return the location and the name of the file
	// so that user can download it
	echo $fileName;
?>
```

Kod 31. PHP skripta *psuGenerator.php*

5. Rezultati

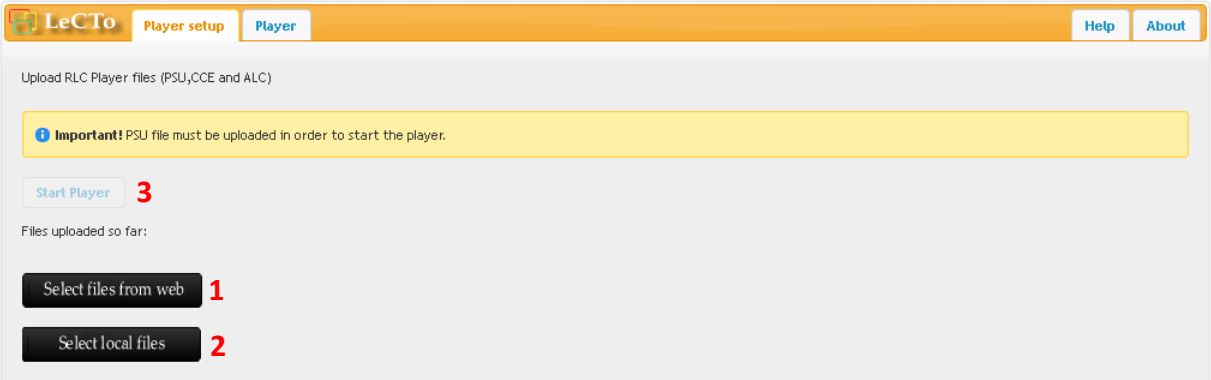
Rezultat projekta je alat za reprodukciju obogaćenih video snimki, LeCTo Player. Na slici 25 prikazan je primjer reprodukcije obogaćene video snimke koristeći alat LeCTo Player. Uz alat LeCTo Player, definirane su i strukture XML datoteka (ALC, PSU i CCE) koje će se generirati koristeći alat LeCTo Recorder.



Slika 25. Reprodukcija obogaćene video snimke predavanja korištenjem alata LeCTo Player

5.1. Način korištenja

Alat LeCTo Player se pokreće otvaranjem *LeCTo.html* stranice u web pregledniku. Alat LeCTo Player ispravno radi u *Internet Explorer* i *Mozilla Firefox* web preglednicima. Otvaranjem *LeCTo.html* web stranice pokreće se aplikacija LeCTo Player. Korisniku će biti prikazan „*Player setup*“ dio aplikacije. U ovom dijelu, korisnik učitava željene ulazne datoteke te pokreće reprodukciju video snimke.

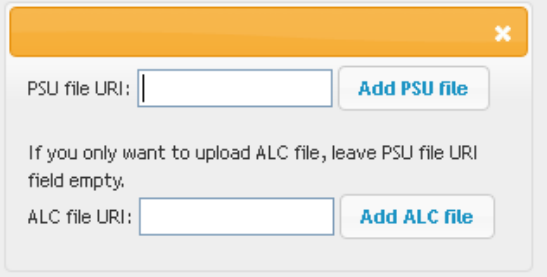


Slika 26. *Player setup* dio aplikacije LeCTo Player

Slika 26 prikazuje *Player setup* dio aplikacije. Gumb označen brojem jedan pokreće postupak učitavanja ulaznih datoteka putem URL adrese. Gumb označen brojem dva pokreće postupak učitavanja ulaznih datoteka s lokalnog diska. Gumb označen brojem tri je gumb *Start Player* za pokretanje video snimke. Gumb ne može biti pritisnut dok PSU i CCE datoteke nisu učitane.

5.1.1. Učitavanje datoteka korištenjem URL poveznice

Pritiskom na gumb *Select files from web* otvara se forma za unos URL poveznica.



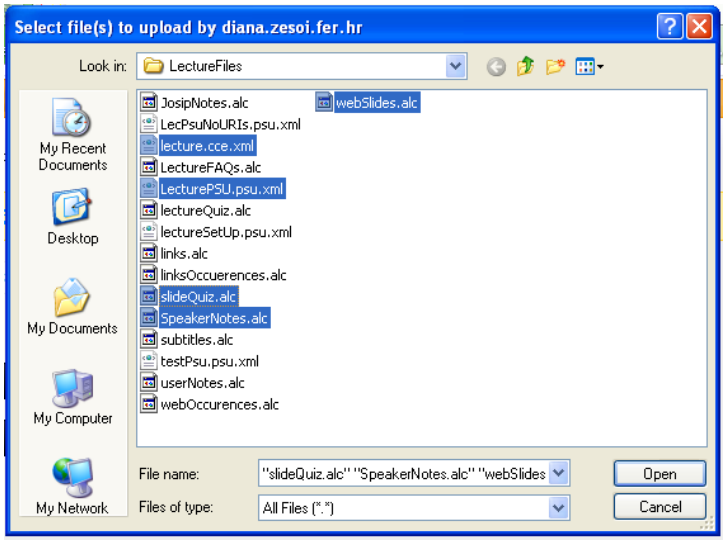
Slika 27. Forma za unos URL poveznica

Forma za unos URL poveznica sadrži dva polja za unos teksta. Prvo polje je za unos URL adrese PSU datoteke, a drugo polje je za unos URL adrese ALC datoteke. Kada se upiše željena URL adrese ulazne datoteke, potrebno je pritisnuti gumb s desne strane polja (*Add PSU file* ili *Add ALC file*). Pritiskom na gumb za dodavanje datoteke, datoteka će se u listu učitanih datoteka dodati te ispisati ime dodane datoteke. Ako korisnik želi učitati više od jedne ALC datoteke, forma se mora više puta pozvati pritiskom na gumb *Select files from web.*

Slika 28. Učitavanje ulaznih datoteka koristeći URL poveznicu

5.1.2. Učitavanje datoteka s lokalnog diska

Pritiskom na gumb *Select local files* otvara se forma za odabir datoteka s lokalnog diska klijentskog računala. Forma za odabir datoteka omogućuje odabir više datoteka odjednom.



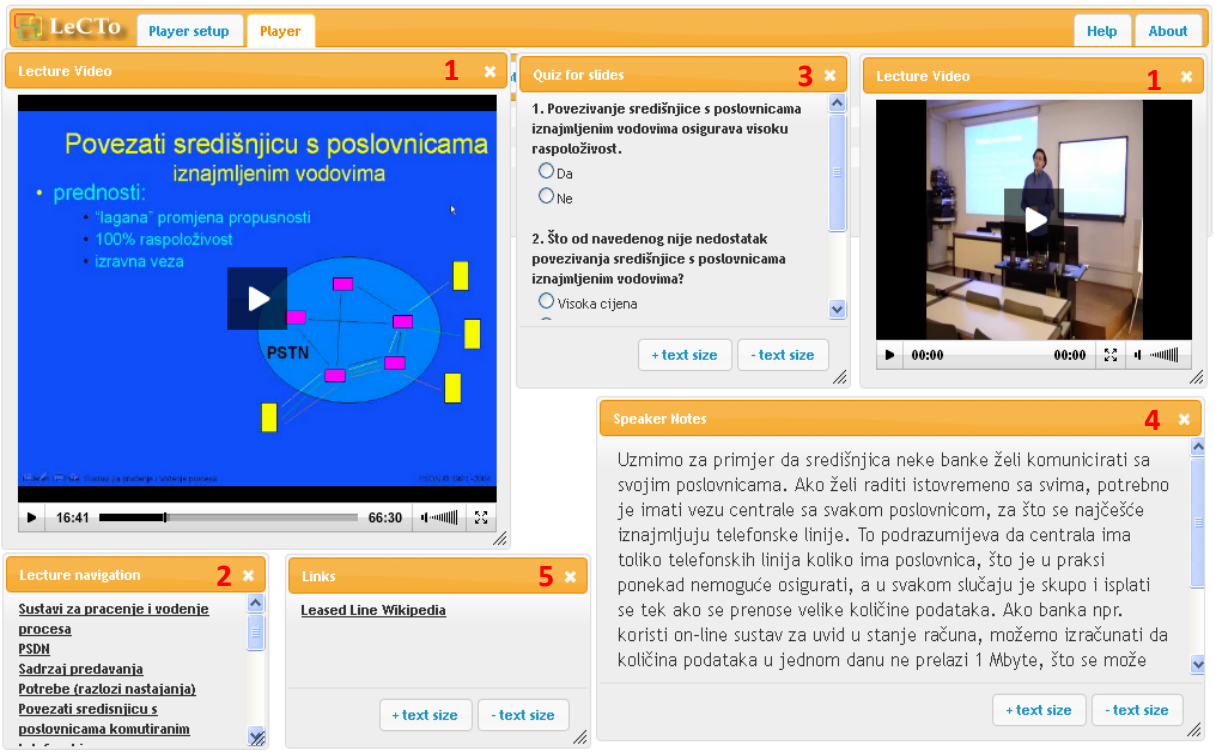
Slika 29. Forma za odabir datoteka s lokalnog diska

Nakon što su datoteke odabrane, lista učitanih datoteka se osvježava te prikazuje nazive novih učitanih datoteka. Isto kao i kod učitavanja datoteka koristeći URL adrese, PSU i CCE datoteke moraju biti učitane kako bi se snimka mogla početi reproducirati. Ako su u PSU datoteci navedene URL adrese svih ostalih datoteka (CCE i ALC) tada je dovoljno samo učitati PSU datoteku.

Dva načina unosa datoteka, putem URL adrese i s lokalnog diska, mogu se kombinirati. Korisnik može učitati nekoliko datoteka koristeći URL adrese, pa onda nekoliko s lokalnog diska.

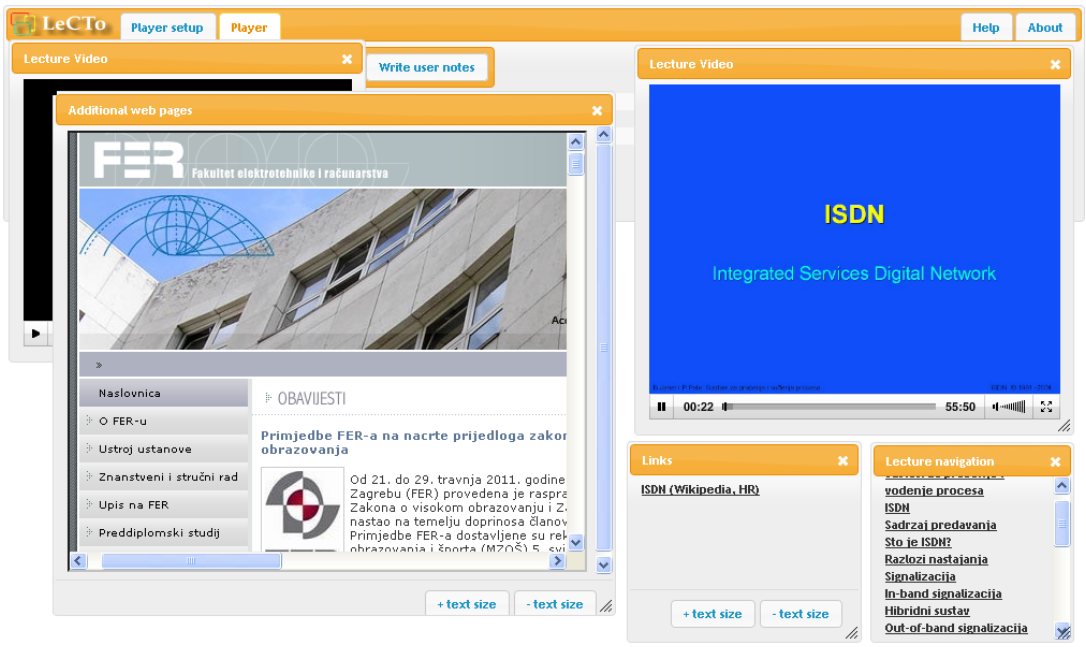
5.1.3. Rad s aplikacijom LeCTo Player

Nakon što su PSU i CCE datoteke učitane, snimka se može početi reproducirati. Snimka se pokreće pritiskom na gumb *Start Player*. Gumb će postati aktivan tek kada su CCE i PSU datoteke učitane. Pokretanjem snimke otvara se *Player* kartica (eng. *tab*).



Slika 30. *Player* dio aplikacije LeCTo Player

Na slici 30 brojevima su označene različite komponente vidljive za vrijeme reproduciranja video snimke predavanja. Brojevima jedan su označeni dijalozi u kojima se nalazi *video player* komponenta zadužena za reproduciranje video snimke. Svaka komponenta reproducira jednu video snimku. Brojem dva označen je navigacijski dijalog. U navigacijskom dijalogu su navedeni nazivi glavnih cjelina prezentacije. Pritiskom na jedna od naziva, snimka se prebacuje na dio gdje je prikazana ta cjelina. Brojevima tri, četiri i pet označeni su dijalozi za prikazivanje dodatnog sadržaja. U dijalogu označenom brojem tri nalaze se pitanja i ponuđeni odgovori. U dijalogu označenom brojem četiri nalazi se tekstualni dodatni sadržaj. Dijalog označen brojem pet sadrži podatkovne poveznice.



Slika 31. Snimka koja sadrži web stranicu kao dodatni sadržaj

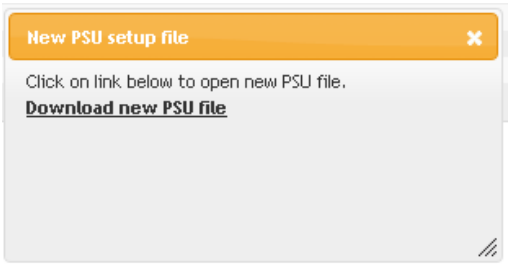
5.1.4. Korisničke kontrole

Slika 32. Korisničke kontrole

Player dio sadrži korisničke kontrole koje omogućuju korisnicima: ponovno pokretanje snimke, stvaranje PSU datoteke s trenutnim postavkama, stvaranje permutiranog indeksa i pisanje korisničkih bilješki. Korisničke kontrole prikazane su na slici 32.

Stvaranje PSU datoteke

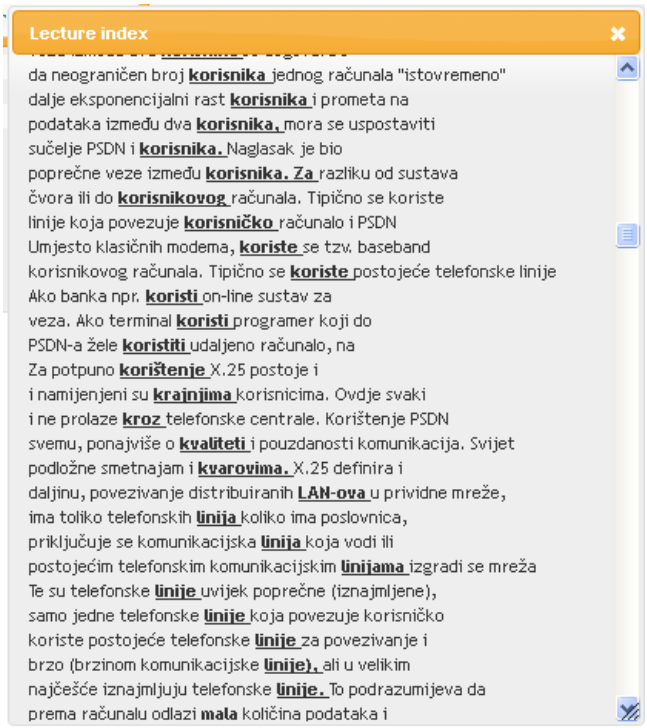
Pritiskom na gumb *Create PSU file* stvara se nova PSU datoteka s trenutnim razmještajem i veličinom dijaloga. Dijalozi koji su zatvoreni kada se pritisne gumb za stvaranje PSU datoteke, biti će izostavljeni u novoj PSU datoteci. Pritiskom na gumb *create PSU file* otvara se dijalog pomoću kojeg se može preuzeti nova PSU datoteka.



Slika 33. Dijalog za preuzimanje nove PSU datoteke

Stvaranje permutiranog indeksa

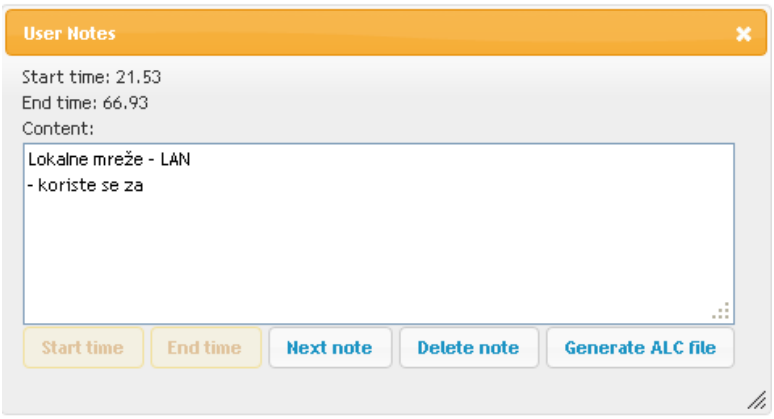
Pritiskom na gumb *Create indeks* stvara se permutirani indeks. Permutirani indeks služi za navigaciju po snimci. Svaka ključna riječ permutiranog indeksa je poveznica koja prebacuje snimku na određeni dio.



Slika 34. Dijalog s permutiranim indeksom

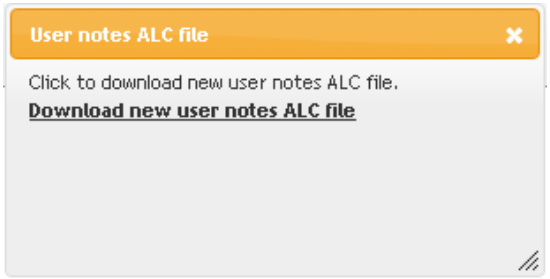
Pisanje korisničkih bilješki

Pritiskom na gumb *Write user notes* otvara se dijalog za pisanje korisničkih bilješki.



Slika 35. Dijalog za pisanje korisničkih datoteka

Kada je korisnik gotov s pisanjem vlastitih bilješki, pritiskom na gumb *Generate ALC file*, generira se ALC datoteka koja sadrži upravo napisane bilješke.



Slika 36. Dijalog za preuzimanje generirane ALC datoteke s korisničkim bilješkama

Zaključak

Alat LeCTo Player omogućuje prikaz obogaćenih snimki predavanja. Zbog inovativnog načina prikazivanja dodatnog sadržaja, snimka predavanja znatno dobiva na kvaliteti. Snimku predavanja, predavač može obogatiti raznim dodatnim sadržajima te tako približiti problematiku predavanja studentima. Dodavanjem web stranica kao dodatnog sadržaja, predavač ili bilo koji autor dodatnog sadržaja nema ograničenja prilikom stvaranja dodatnog sadržaja. Gledajući snimku, gledatelj može pisati vlastite komentare. Korisnički komentari spremaju se u ALC format te na taj način gledatelji (studenti) mogu izmjenjivati svoje bilješke.

Alat LeCTo Player izrađen je korištenjem standardiziranih web tehnologija koje podražavaju svjetski najkorišteniji Internet preglednici i operacijski sustavi. Zbog toga ala LeCTo može koristiti široki spektar korisnika.

Budućnost alata LeCTo Player je dovršavanje nedovršenih funkcija te konstantno unaprjeđivanje i razvijanje novih mogućnosti. Prvi koraci u budućnosti alata LeCTo Player su: dodavanje mogućnosti RTMP prijenosa video snimki, unaprjeđenje sučelja za pisanje korisničkih bilješki i prilagodba alata LeCTo Player za rad na Internet preglednicima *Google Chrome* i *Safari*.

Iako još postoje neke nedovršene funkcije, alat LeCTo Player je potpuno funkcionalan i može se koristiti kao e-learning alat pri raznim edukacijskim ustanovama.

Reference

[1]. Web izvor: E-Učenje sad
URL: http://www.fer.hr/e-ucenje_sad/e-knjige/web_stranice/microsoft_producer
Pristup: 6. lipnja, 2011.

[2]. Web izvor: iPresent
URL: http://www.ipresent.net/
Pristup: 6. lipnja, 2011.

[3]. Web izvor: MIT OpenCourseWare
URL: http://ocw.mit.edu/index.htm
Pristup: 6. lipnja, 2011.

[4]. Web izvor: Harvard Extension Classroom
URL:
http://cm.dce.harvard.edu/2011/02/23228/L01/seg1/index_FlashSingleHighBandwidth.shtml
Pristup: 6. lipnja, 2011.

[5]. Web izvor: SlideSix
URL: http://slidesix.com/
Pristup: 6. lipnja, 2011.

[6]. Web izvor: *Flash video*
URL: http://en.wikipedia.org/wiki/Flash_Video
Pristup: 6. lipnja, 2011.

[7]. Web izvor: *Application Programming Interface*
URL: http://en.wikipedia.org/wiki/Application_programming_interface
Pristup: 7. lipnja, 2011.

[8]. Web izvor: *Same origin policy*
URL: http://en.wikipedia.org/wiki/Same_origin_policy
Pristup: 13. lipnja, 2011.

[9]. Web izvor: Službena dokumentacija *Uploadify* dodatka
URL: http://www.uploadify.com/documentation/
Pristup: 13. lipnja, 2011.

[10]. Web izvor: Jquery *User Interface*
URL: http://jqueryui.com/demos/
Pristup: 13. lipnja, 2011.

Literatura

1. Web sadržaj: *Real Time Messaging Protocol*
   URI: http://en.wikipedia.org/wiki/Real_Time_Messaging_Protocol
   Pristup: 7. lipnja, 2011.
   Zadnja promjena: 1. lipnja, 2011. u 08:33 sati
2. Web sadržaj: w3Schools: Javascript
   URL: http://www.w3schools.com/js/default.asp
   Pristup: 7. lipnja, 2011.
3. Web sadržaj: PHP
   URL: http://www.php.net/
   Pristup: 7. lipnja, 2011.
4. Web sadržaj: jwplayer
   URL: http://www.longtailvideo.com/players/
   Pristup: 7. lipnja, 2011.
5. Web sadržaj: uploadify
   URL: http://www.uploadify.com/
   Pristup: 7. lipnja, 2011.
6. Web sadržaj: web domena
   URL: http://en.wikipedia.org/wiki/Domain_name
   Pristup: 13. lipnja, 2011.
7. Web sadržaj: AJAX
   URL: http://en.wikipedia.org/wiki/Ajax_(programming)
   Pristup: 13. lipnja, 2011.
8. Web sadržaj: XML DOM
   URL: http://dhillman.com/theplace/xml/xmldomtech.htm
   Pristup: 13. lipnja, 2011.

9. Web sadržaj: *XML DOM methods*
   URL: http://msdn.microsoft.com/en-us/library/ms757828(v=vs.85).aspx
   Pristup: 13. lipnja, 2011.
10. Jquery *User Interface*
   URL: http://jqueryui.com/
   Pristup: 13. lipnja, 2011.

Sažetak

On-line prikaz obogaćenih snimki predavanja

Cilj obrazovnih ustanova, ali i studenata je što kvalitetniji prijenos znanja. Zbog toga obrazovne ustanove sve češće snimke svojih predavanja prezentiraju na vlastitim mrežnim stranicama. Cilj ovog rada je na temelju analize svojstava nekih od postojećih sustava za prikaz video snimki izraditi alat za on-line prikaz obogaćenih video snimki.

Izrađeni alat je alat LeCTo Player. Alat LeCTo Player omogućuje on-line dvije paralelne i sinkronizirane video snimke. Snimke su obogaćene dodatnim sadržajem koji je sinkroniziran sa snimkom predavanja. Dodatni sadržaj za neku snimku može stvoriti bilo tko. Dodatni sadržaj kojeg podržava alat LeCTo Player je: tekstualni dodatni sadržaj, podatkovne poveznice, pitanja i odgovori i web stranice. Autor dodatnog sadržaja može odrediti točne vremenske trenutke kada će određeni dodatni sadržaj biti prikazan i kada će nestati.

U ovom radu prikazan je proces implementacije alata LeCTo Player.

Ključne riječi*:* e-learning, LeCTo Player, snimke predavanja, obogaćene snimke predavanja, sinkronizacija video snimki.

Summary

Online preview of enriched lecture recordings

The goal of educational institutions and students is a quality knowledge transfer. Because of that goal educational institutions are often publishing their lecture recordings on their web sites. The aim of this thesis is to analyze properties of some of the existing lecture recording preview tools. Using that analysis a tool for online preview of enriched lecture recordings should be built.

Implemented tool is LeCTo Player. LeCTo Player enables online preview of two parallel and synchronized lecture recordings. Lecture recordings are enriched with additional lecture content which is synchronized with the lecture recording. The author of additional lecture content can be anybody. LeCTo Player supports: textual additional content, links, questions and answers and web pages. The author of additional lecture content can define exact time periods when the content will appear and disappear.

This thesis shows the implementation process of the LeCTo Player.

Keywords: e-learning, LeCTo Player, lecture recordings, enriched lecture recordings, video recordings synchronization

Privitak

Instalacija alata LeCTo Player

Da bi se alat LeCTo Player instalirao potreban je poslužitelj koji može pokretati PHP skripte. Budući da je LeCTo Player implementiran koristeći HTML i web tehnologije (*jquery, javascript, AJAX*) dovoljno je samo kopirati datoteku LeCTo Player na poslužitelj. Alatu LeCTo Player će se tada pristupati koristeći poveznicu:

```
http://www.domenaPoslužitelja/LeCTo%20Player/LeCTo.html
```

Direktoriji `LeCToPlayer/files` i `LeCToPlayer/files/tempFiles` moraju imati omogućeno pisanje (*chmod* na 777).