



**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**



**УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
НОВИ САД
Департман за рачунарство и аутоматику
Одсек за рачунарску технику и рачунарске комуникације**

ЗАВРШНИ (BACHELOR)РАД

Кандидат: Крсто Лазић
Број индекса: 12658

Тема рада: Реализација протокола за динамички адаптиван пренос података (MPEG DASH)

Ментор рада: проф. др Никола Теслић

Нови Сад, јул, 2012



УНИВЕРЗИТЕТ У НОВОМ САДУ ● ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

| | |
|---|---|
| Редни број, РБР: | |
| Идентификациони број, ИБР: | |
| Тип документације, ТД: | Монографска документација |
| Тип записа, ТЗ: | Текстуални штампани материјал |
| Врста рада, ВР: | Завршни (Bachelor) рад |
| Аутор, АУ: | Крсто Лазић |
| Ментор, МН: | проф. др Никола Теслић |
| Наслов рада, НР: | Реализација протокола за динамички адаптиван пренос података (MPEG-DASH) |
| Језик публикације, ЈП: | Српски / латиница |
| Језик извода, ЈИ: | Српски |
| Земља публиковања, ЗП: | Република Србија |
| Уже географско подручје, УГП: | Војводина |
| Година, ГО: | 2012 |
| Издавач, ИЗ: | Ауторски репринт |
| Место и адреса, МА: | Нови Сад; трг Доситеја Обрадовића 6 |
| Физички опис рада, ФО: (поглавља/страна/ цитата/табела/слика/графика/прилога) | 7/36/0/0/7/0/0 |
| Научна област, НО: | Електротехника и рачунарство |
| Научна дисциплина, НД: | Рачунарска техника |
| Предметна одредница/Кључне речи, ПО: | Пренос података, мултимедија |
| УДК | |
| Чува се, ЧУ: | У библиотеци Факултета техничких наука, Нови Сад |
| Важна напомена, ВН: | |
| Извод, ИЗ: | У овом раду приказан је један од начина имплементације ISO "Live" профила MPEG-DASH стандарда за динамички адаптиван пренос података преко HTTP протокола |
| Датум прихватања теме, ДП: | |
| Датум одбране, ДО: | |
| Чланови комисије, КО: | Председник: др. Јелена Ковачевић |
| | Члан: |
| | Члан, ментор: проф. др Никола Теслић |
| | Потпис ментора |



KEY WORDS DOCUMENTATION

| | |
|---|--|
| Accession number, ANO : | |
| Identification number, INO : | |
| Document type, DT : | Monographic publication |
| Type of record, TR : | Textual printed material |
| Contents code, CC : | Bachelor Thesis |
| Author, AU : | Krsto Lazić |
| Mentor, MN : | PhD Nikola Teslić |
| Title, TI : | Realization of dynamic adaptive streaming protocol (MPEG-DASH) |
| Language of text, LT : | Serbian |
| Language of abstract, LA : | Serbian |
| Country of publication, CP : | Republic of Serbia |
| Locality of publication, LP : | Vojvodina |
| Publication year, PY : | 2012 |
| Publisher, PB : | Author's reprint |
| Publication place, PP : | Novi Sad, Dositeja Obradovica sq. 6 |
| Physical description, PD : <small>(chapters/pages/ref./tables/pictures/graphs/appendixes)</small> | 7/36/0/0/7/0/0 |
| Scientific field, SF : | Electrical Engineering |
| Scientific discipline, SD : | Computer Engineering, Engineering of Computer Based Systems |
| Subject/Key words, S/KW : | Streaming, Multimedia |
| UC | |
| Holding data, HD : | The Library of Faculty of Technical Sciences, Novi Sad, Serbia |
| Note, N : | |
| Abstract, AB : | One implementation of ISO "Live" profile of MPEG-DASH standard for dynamic adaptive streaming over HTTP protocol is described in this paper. |
| Accepted by the Scientific Board on, ASB : | |
| Defended on, DE : | |
| Defended Board, DB : | President: |
| | Member: |
| | Member, Mentor: |
| | Menthor's sign |

SADRŽAJ

| | |
|--|----|
| 1. Uvod..... | 1 |
| 2. Teorijske osnove..... | 4 |
| 2.1 Opis medija prezentacije (MPD)..... | 6 |
| 2.2 Format segmenta | 7 |
| 2.3 Profili..... | 7 |
| 2.3.1 ISO media „Live“ profil..... | 8 |
| 2.3.1.1 Ograničenja dokumenta za opis medija prezentacije..... | 9 |
| 2.3.1.2 Ograničenja formata segmenta | 9 |
| 3. Koncept rešenja | 10 |
| 3.1 DASH klijent..... | 10 |
| 3.2 Objekti rešenja | 11 |
| 3.2.1 Rukovalac MPD dokumentom | 12 |
| 3.2.1.1 Dobavljač MPD dokumenta | 13 |
| 3.2.1.2 Parser MPD dokumenta | 14 |
| 3.2.2 Rukovalac HTTP protokolom..... | 14 |
| 3.2.3 Rukovalac segmentima..... | 14 |
| 3.2.3.1 Kreiranje listi segmenata | 15 |
| 3.2.3.2 Odabir segmenta | 16 |
| 3.2.3.3 Adaptaciona logika | 16 |
| 3.2.4 Modul za reprodukciju medija sadržaja | 18 |
| 4. Programsko rešenje..... | 19 |
| 4.1 Rukovalac MPD dokumentom..... | 19 |
| 4.2 Rukovalac HTTP protokolom..... | 21 |

| | | |
|-----|--|----|
| 4.3 | Modul za reprodukciju medija sadržaja..... | 22 |
| 4.4 | Rukovalac segmentima..... | 24 |
| 5. | Ispitivanje i verifikacija | 27 |
| 6. | Zaključak..... | 28 |
| 7. | Literatura..... | 29 |

SPISAK SLIKA

| | |
|---|----|
| Slika 2.1 Scenario prenosa podataka između HTTP poslužioca i DASH klijenta..... | 5 |
| Slika 2.2 Hijerarhijska struktura model podataka u MPD dokumentu | 6 |
| Slika 2.3 Profili MPEG-DASH standarda..... | 8 |
| Slika 3.1 Model rješenja dinamičkog adaptivnog prenosa podataka..... | 12 |
| Slika 3.2 Rukovalac MPD dokumentom..... | 13 |
| Slika 3.3 Rukovalac segmentima..... | 15 |
| Slika 3.4 Algoritam adaptacione logike..... | 17 |

SKRAĆENICE

| | |
|-------------|--|
| DASH | - <i>Dynamic Adaptive Streaming over HTTP</i> , dinamički adaptivan prenos podataka preko HTTP protokola |
| HTTP | - <i>Hypertext Transfer Protocol</i> , protokol za prenos hiper teksta |
| ISO | - <i>International Organisation for Standardization</i> , međunarodna organizacija za standarde |
| MPEG | - <i>Motion Picture Expert Group</i> |
| MPD | - <i>Media Presentation Description</i> , opis medija prezentacije |
| RTP | - <i>Real-time Transport Protocol</i> |
| URI | - <i>Uniform Resource Identifier</i> , jednoobrazni identifikator resursa |
| URL | - <i>Uniform Resource Locator</i> , jedinstvena adresa resursa |
| XML | - <i>Extensive Markup Language</i> , proširivi metajezik za označavanje tekstualnih dokumenata |

1. Uvod

U ovom radu biće predstavljen jedan od načina implementacije profila koji je optimizovan za kompresiju uživo i prenos segmenata sa malim kašnjenjem (ISO „Live“ profil)[1], gdje se svaki segment sastoji od jednog video isječka u ISO formatu [2] i relativno male je dužine. Akcenat je stavljen na obezbjeđivanje podrške za efikasno preključivanje između video segmenata različitih bitskih brzina kao i obezbjeđivanje adaptacione logike kako bi se omogućila kontinualna reprodukcija video i audio toka. Rješenje je implementirano na operativnom sistemu Linux u programskom jeziku C++ sa naglaskom da rješenje bude nezavisno od fizičke arhitekture i sistemske programske podrške kako bi se obezbijedila lakša prenosivost rješenja na različite platforme.

Dostavljanje video sadržaja preko interneta započelo je devedesetih godina prošlog vijeka i na samom početku kao glavni izazovi postavili su se vremenski zahtjevi i velika količina podataka koju je potrebno dostaviti. Kao odgovor na ove zahtjeve dizajniran je protokol (*Real-time Transport Protocol – RTP*) koji definiše formate audio i video paketa, kao i rukovođenje sesijom koja omogućava prenos ovih paketa sa malom količinom redundantnih podataka. RTP se pokazao kao dobar protokol za upravljane mreže (*engl. Managed networks*). Međutim, u današnjem internetu ova vrsta mreža zamijenjena je mrežama za dostavu sadržaja (*engl. Content Delivery Networks*) od kojih mnoge ne podržavaju prenos video i audio toka preko RTP protokola [3]. Uz to, RTP paketima nije dozvoljen prolaz kroz mnoge sisteme zaštite u mreži. Konačno, prenos video i audio toka preko RTP protokola zahtijeva da poslužilac rukovodi posebne sesije za svakog od klijenata što čini širokopoljasne dostave veoma zahtjevnim i intenzivnim po pitanju resursa.

Sa povećanjem protoka u internetu, značaj dostavljanja video i audio podataka u malim paketima je umanjen. Multimedijalni sadržaji sada mogu biti efikasno dostavljeni u većim dijelovima korišćenjem HTTP protokola[4].Prenos video i audio toka preko HTTP protokola ima nekoliko prednosti.Prvo, infrastruktura interneta je napredovala dovoljno da može efikasno da podrži HTTP. Na primjer, mreže za dostavu sadržaja obezbjeđuju lokalno privremenočuvanje podataka (*engl. caching*), što smanjuje saobraćaj na duge razdaljine. Takođe, HTTP je prijateljski nastrojen prema sistemima zaštite u mreži.Tehnologija HTTP servera je efikasna po pitanju cijene, što prenos video i audio toka milionima korisnika čini isplativim. Drugo, kod HTTP prenosa toka podataka, klijent rukovodi prenos bez potrebe za održavanjem sesije na strani poslužioca. Samim tim, opsluživanje velikog broja klijenata ne predstavlja dodatni trošak.

Ipak, prenos video i audio sadržaja preko interneta još uvijek nije dostigao potencijalni nivo razvoja. Jedan razlog je to što je danas svaka komercijalna platforma zatvoren sistem sa svojim formatima sadržaja i protokola prenosa toka podataka. Drugim riječima, ne postoji interoperabilnost između uređaja i poslužilaca različitih proizvođača. Nedavno obavljene studije predviđaju da će u skorij budućnosti video sadržaj predstavljati većinu internet saobraćaja. Kako bi ovo bilo omogućeno, potrebno je da se usvoji standard koji obezbjeđuje interoperabilnost između različitih poslužilaca i uređaja. Dostizanje takve interoperabilnosti je odinstrumentalnog značaja za rast tržišta, jer zajednički ekosistem sadržaja i usluga će moći da snabdijeva širok pojas uređaja kao što su personalni računari, televizori i drugi prijemnici digitalnog televizijskog signala (*engl. set-top boxes*), prenosivi računari, igračke konzole i mobilni telefoni. Protokol za dinamički adaptivan prenos podataka MPEG-DASH razvijen je upravo radi toga.

MPEG-DASH je tehnologija za dinamički adaptivni prenos podataka predložena od strane MPEG radne grupe i trenutno se nalazi u fazi verifikacije od strane ISO organizacije za standarde. Namjena ove tehnologije je da podrži model prenosa toka podataka u kom se kontrola sesije nalazi isključivo na klijentskoj strani. Klijenti zahtijevaju podatke putem HTTP protokola od standardnih poslužilaca koji ne moraju da budu svjesni specifičnosti MPEG-DASH tehnologije. Samim tim, ovaj standard nije usmeren ka procedurama klijenta i poslužioca, nego ka formatima podataka koji se koriste da bi se obezbijedio dinamički adaptivan prenos.

U drugom poglavlju rada date su teorijske osnove MPEG-DASH standarda koje su vezane za profil koji je potrebno implementirati. Treće poglavlje opisuje koncept rješenja gdje je počevši od opisa opšte slike rješenja razrađeno do nivoa detalja kako bi rješenje trebalo da izgleda. U četvrtom poglavlju data je konkretizacija rješenja koja podrazumijeva detalje

realizacije modula sa opsimi sprege između njih. U petom i šestom poglavlju je dat kratak osvrt na ono što je urađeno sa evaluacijom rada rješenja.

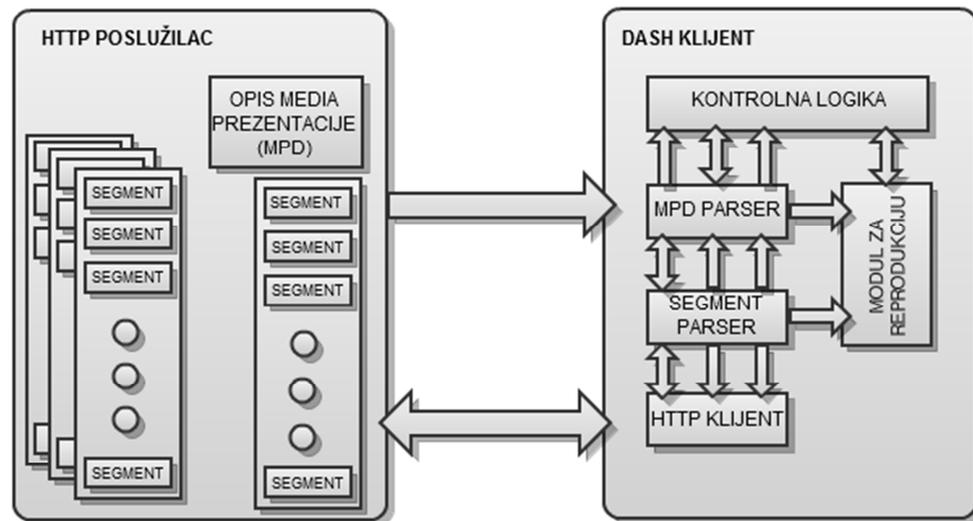
2. Teorijske osnove

Dinamički adaptivan prenos podataka preko HTTP protokola (DASH) specificira XML i binarne formate koji omogućavaju dostavljanje video i audio toka od standardnih HTTP poslužilaca HTTP klijentima. Standardom su prvenstveno definisana dva formata:

- MPD - dokument koji opisuje prezentaciju medija sadržaja (audio, video, tekst...). Ovim dokumentom se definišu formati po kojima se stvaraju identifikatori resursa, kao i kontekst u kome se oni nalaze unutar medija prezentacije. Pomenuti identifikatori resursa su u stvari HTTP-URL, po mogućnosti kombinovani sa bajt opsezima.
- Formati segmenta specificiraju formate tijela HTTP odgovora. Segmenti tipično sadrže efikasno kodovane medija podatke i metapodatke u skladu sa opštim medija formatima.

Veoma je važno da se naglasi da MPEG-DASH standard ne specificira način dobavljanja MPD dokumenta, formate kodovanja i dekodovanja sadržaja, kao ni ponašanje klijenta prilikom dobavljanja sadržaja, adaptacionu heuristiku ili način reprodukcije sadržaja. Standard je prije svega namijenjen obezbjeđivanju interoperabilnosti između uređaja i poslužilaca različitih proizvođača.

Na slici 2.1 prikazan je jednostavan scenario prenosa toka medija podataka između HTTP poslužioca i DASH klijenta.



Slika 2.1 Scenario prenosa podataka između HTTP poslužioca i DASH klijenta

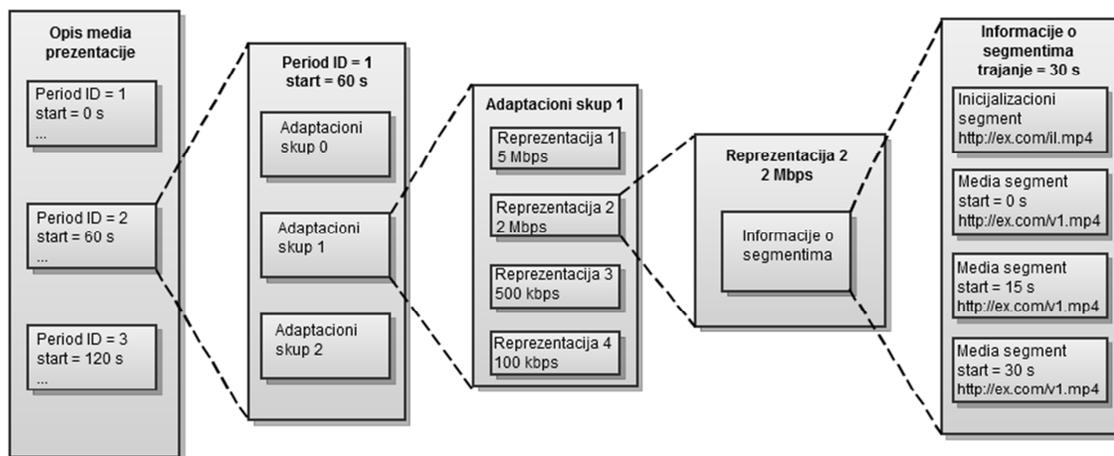
Multimedijalni sadržaj se proizvodi i skladišti na strani poslužioca i prenosi se putem HTTP protokola. Sadržaj se na strani poslužioca sastoji iz dva dijela: opisa medija prezentacije (MPD), koji opisuje dostupni sadržaj, njegove različite alternative, URL adrese i druge karakteristike; i segmenata, koji sadrže tok medija podataka u formi skupa disjunktih dijelova u jednoj ili više datoteka.

Kako bi započeo reprodukciju sadržaja DASH klijent prvo mora da dobavi MPD dokument. Parsiranjem MPD dokumenta klijent saznaje informacije o dostupnosti medija sadržaja, tipovima sadržaja, vremenskim ograničenjima, rezolucijama, minimalnim i maksimalnim propusnim opsezima, postojanju različitih enkodovanih alternativa medija komponenti, lokacijama sadržaja u mreži i drugim karakteristikama. Pomoću ovih informacija DASH klijent odabira odgovarajuću enkodovanu alternativu i započinje dobavljanje toka podataka HTTP GET zahtjevima.

Nakon odgovarajućeg prikupljanja podataka u memoriju (*engl. buffering*) koje je neophodno kako bi se umanjio uticaj varijacija protoka u mreži, klijent nastavlja dobavljanje sljedećih segmenata, a pri tom vrši i posmatranje fluktuacija propusnog opsega mreže. Zavisno od izmjerenih promjena u propusnom opsegu i pomoću adaptacione logike klijent odlučuje da li će nastaviti sa dobavljanjem segmenata iste bitske brzine ili neke od dostupnih bitskih brzina koja više odgovara stanju propusnog opsega u mreži.

2.1 Opis medija prezentacije (MPD)

Dinamički prenos podataka preko HTTP protokola zahtijeva da na strani poslužioca budu dostupne različite alternative bitskih brzina multimedijalnog sadržaja. Osim toga multimedijalni sadržaj može da se sastoji iz više komponenti (npr. audio, video i tekst) od kojih svaka može da ima različite karakteristike. MPEG-DASH standardom je predviđeno da te karakteristike budu opisane u XML dokumentu koji se zove opis medija prezentacije (MPD). Slika 2.2 pokazuje hijerarhijsku strukturu modela podataka u ovom dokumentu.



Slika 2.2 Hijerarhijska struktura model podataka u MPD dokumentu

MPD se sastoji od jednog ili više perioda koji predstavljaju vremenski interval na vremenskoj osi. Svaki period ima vrijeme početka i trajanje i sastoji se od jednog ili više adaptacionih skupova. Adaptacioni skup daje informacije o jednoj ili više medija komponenti i njihovim različitim kodovanim alternativama. Na primjer, može da sadrži bitske brzine video komponenti istog sadržaja. Svaki adaptacioni skup uglavnom sadrži više reprezentacija.

Reprezentacija predstavlja kodovanu alternativu iste medija komponente, koja se od ostalih reprezentacija razlikuje po bitskoj brzini, rezoluciji, broju kanala i drugim karakteristikama. Svaka reprezentacija se sastoji iz više segmenata. Segmenti su vremenski složeni dijelovi toka medija podataka. Svaki segment posjeduje URI, tj. adresabilnu lokaciju na strani poslužioca čiji sadržaj može biti zahtijevan HTTP GET naredbom.

Da bi koristio ovaj model podataka, DASH klijent mora prvo da parsira MPD XML dokument. Nakon parsiranja klijent na osnovu opisanih elemenata iz MPD-a, klijentovih mogućnosti i zahtjeva korisnika selektuje skup reprezentacija koje će da koristi. Svaki opis reprezentacije sadrži informacije o segmentima, što omogućuje formulisanje HTTP zahtjeva za svaki segment, opcionalno sa bajt opsegom. Za reprezentacije čiji se sadržaj prenosi uživo, MPD

nudi i vrijeme početka i kraja dostupnosti segmenta, kao i fiksnu ili promjenljivu dužinu segmenta.

MPD dokument nudi dovoljnu količinu informacija klijentu kako bi on korisniku mogao da pruži uslugu reprodukcije toka medija podataka pristupanjem segmentima preko jednog od protokola specificiranih standardom. U daljem tekstu biće smatrano da je taj protokol HTTP/1.1.

2.2 Format segmenta

Multimedijalnom sadržaju se može pristupiti kao kolekciji segmenata. Segment je definisan kao tijelo odgovora na klijentov HTTP GET zahtjev. Medija komponenta je kompresovana i podijeljena na više segmenata. Prvi segment može biti tzv.inicijalizacioni segment i on sadrži potrebne informacije kako bi se inicijalizovao dekodier na klijentskoj strani i ne sadrži nikakve medija podatke.

Tok medija podataka je zatim podijeljen u više uzastopnih medija segmenata. Svakom medija segmentu pridružen je jedinstveni URL (po mogućnosti sa bajt opsegom), indeks i eksplicitno ili implicitno vrijeme početka i trajanje. Svaki medija segment sadrži najmanje jednu pristupnu tačku odakle dekodovanje može da počne koristeći samo podatke koji dolaze poslije nje.

MPEG-DASH definiše kontejnere segmenata za ISO format kao i za MPEG-2 TS, ne vodi računa o kodeku i podržava kako multipleksirani tako i nemultipleksirani kodovani sadržaj.

2.3 Profili

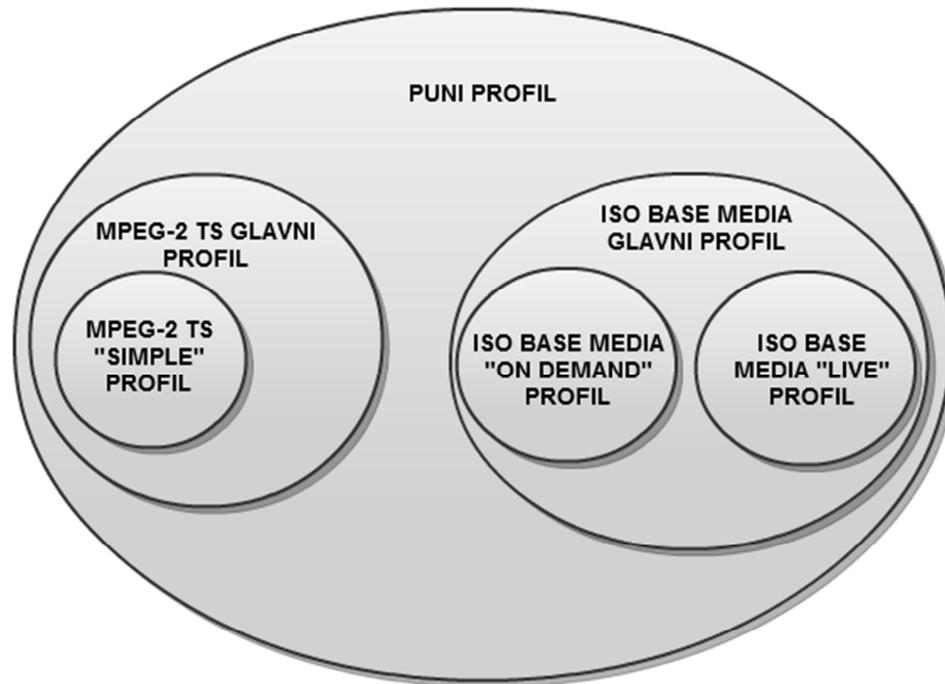
Profili dinamičkog adaptivnog prenosa podataka su definisani da bi se omogućila interoperabilnost i kako bi se istaklo koje osobine je potrebno da posjeduje konkretna implementacija.

Profil nameće skup određenih ograničenja. Ta ograničenja se tipično odnose na osobine dokumenta za opis medija prezentacije (MPD) i formate segmenata, ali takođe mogu da se odnose i na sadržaj koji se dostavlja kroz segmente, kao što je na primjer tip medija sadržaja, format medija sadržaja, algoritam kompresije, ili na kvantitativne mjere kao što je bitska brzina, trajanje i veličina segmenta, itd.

Profil se takođe može razumjeti i kao dozvola DASH klijentu da implementira samo one osobine koje su definisane profilom. Ipak, ponašanje DASH klijenta nije normativno

specificirano MPEG-DASH standardom, tako da nije specificirano ni kako će se DASH klijent prilagoditi profilu. Otuda, profili specificiraju samo ograničenja koja su vezana za MPD dokument i formate segmenata, a na ponašanje DASH klijenta utiču samo indirektno.

Standardom je definisano šest profila i njihov međusobni odnos prikazan je na slici 2.3.



Slika 2.3 Profili MPEG-DASH standarda

2.3.1 ISO media „Live“ profil

Ovaj profil je optimizovan za kompresiju uživo i prenos segmenata sa malim kašnjenjem gdje se segment sastoji od jednog isječka videa u ISO formatu relativno kratkog trajanja. MPD dokument se vrlo često dinamički mijenja tako da ga je potrebno dobavljati periodično sa učestanošću koja je u njemu definisana. Segmenti se mogu zahtijevati kada budu dostupni formiranjem zahtjeva pomoću šablona koji je definisan u MPD dokumentu i u tom slučaju MPD dokument se dobavlja sa manjom učestanošću. Segmenti se mogu spajati na granicama i dekodovati bez ikakvih zavisnosti od drugih segmenata nazavisno od toga kojoj reprezentaciji pripadaju u okviru adaptacionog skupa (nezavisno od bitske brzine). Uprkos tome što je profil optimizovan za servise koji se prenose uživo, može se desiti da MPD dokument bude statički ukoliko se isti sadržaj sačuva kako bi se omogućio kasniji prenos na zahtjev.

2.3.1.1 Ograničenja dokumenta za opis medija prezentacije

Dokument za opis medija prezentacije (MPD) podliježe sljedećim ograničenjima:

- Identifikator profila je URN „urn:mpeg:dash:profile:isoff-live:2011“.
- Šablon za formiranje zahtjeva za segmentima mora biti prisutan bar na jednom od tri nivoa: nivo perioda koji sadrži reprezentaciju, nivo adaptacionog skupa koji sadrži reprezentaciju ili nivo same reprezentacije.
- Na početku svakog segmenta mora da se nalazi pristupna tačka odakle dekodovanje može da počne koristeći samo podatke koji dolaze poslije nje.
- Segmenti moraju biti poravnati tako da ne postoji praznina ni preklapanje između njih.
- Preporučuje se da trajanje segmenata bude relativno kratko.
- Ukoliko postoje adaptacioni skupovi organizovani u podskupove, elementi takvih podskupava se ignorišu.

2.3.1.2 Ograničenja formata segmenta

Reprezentacije i segmenti u ovom profilu podliježu sljedećim ograničenjima:

- Svaka reprezentacija mora da ima jedan inicijalizacioni segment i bar jedan medija segment.

U okviru medija segmenta, svi kontejneri koji sadrže indeksne informacije moraju se nalaziti prije kontejnera koji sadrži fragment videa („moof“).

3. Koncept rešenja

Prilikom projektovanja rješenja zadanog problema poželjno je naći optimalan način modularizacije kako bi se došlo do elegantnog i logički korektnog rješenja koje ostavlja prostora za eventualnu nadogradnju ukoliko se ukaže potreba za tim i čiju valjanost je moguće bez poteškoća ispitati. U ovom rješenju posebna pažnja je posvećena ovoj osobini. Takođe, veliki trud je uložan da rješenje bude što je više moguće nezavisno od fizičke arhitekture i sistemske programske podrške kako bi se obezbijedila prenosivost na različite platforme.

3.1 DASH klijent

DASH klijent je vođen informacijama koje su date u MPD dokumentu. Sljedeći primjer opisuje scenario koji treba da obezbijedi kontinualno dobavljanje toka medija podataka. Prilikom analize problema ovaj scenario je uzet u obzir kako bi lakše bili uočeni objekti rješenja.

- 1) Klijent parsira MPD dokument i zatim selektuje adaptacione skupove koji odgovaraju njegovom okruženju na osnovu informacija ponuđenih u svakom adaptacionom skupu.
- 2) U okviru svakog adaptacionog skupa selektuje jednu određenu reprezentaciju, tipično na osnovu propusnog opsega u mreži ali takođe uzimajući u obzir i sposobnosti dekodovanja i prikazivanja modula za reprodukciju medija sadržaja. Zatim stvara listu segmenata za svaku od reprezentacija, da bi u naprijed bio poznat redoslijed segmenata kao i sve informacije koje su potrebne da bi se formirao HTTP GET zahtjev.

- 3) Klijent na osnovu generisane liste segmenata šalje zahtjeve za segmentima HTTP poslužiocu.
- 4) Klijent baferuje medija podatke najmanje onoliko vremena koliko je predloženo u MPD dokumentu prije nego što počne sa reprodukcijom sadržaja. Nakon identifikacije pristupnih tačaka odakle može da započne dekodovanje medija tokova, klijent započinje reprodukciju sadržaja.
- 5) Kada je reprodukcija započela, klijent nastavlja da konzumira medija sadržaj kontinualnim zahtjevima za medija segmentima od poslužioca. Klijent može da se preključi na drugu reprezentaciju (bitsku brzinu) uzimajući u obzir ažurirane informacije iz MPD dokumenta i iz njegovog okruženja, odnosno, stanje protoka u mreži i mogućnosti modula za reprodukciju medija sadržaja.
- 6) Ako MPD dokument nije statički, klijent ima obavezu da prije zahtjeva za dobavljanjem novog segment provjeri da li je MPD dokument ažuran. Za provjeravanje ažurnosti MPD dokumenta koriste se informacije koje postoje u samom dokumentu. Ukoliko dokument nije ažuran, šalje se zahtjev za dobavljanjem ažurne verzije. Prelazi se na korak 5).

3.2 Objekti rešenja

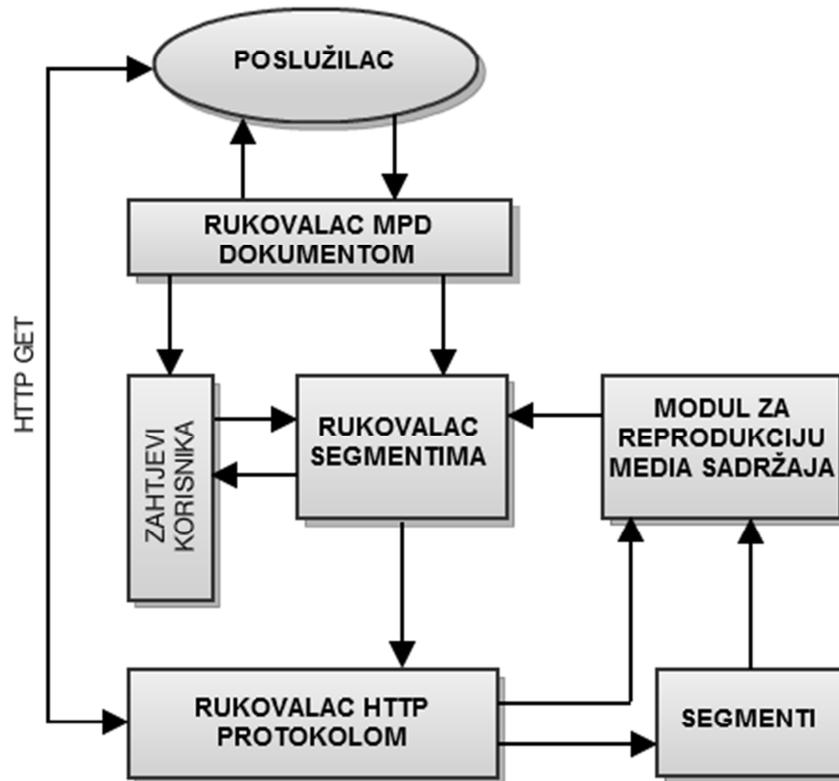
Gledano sa aspekta prenosivosti i nezavisnosti od fizičke arhitekture, prilikom analize problema uočeno je da se jedan modul izdvaja od ostalih po tome što jedini zavisi od fizičke arhitekture, a to je modul za reprodukciju medija sadržaja. Imajući ovo u vidu potrebno je na početku istaći da je od velikog značaja da sprega prema ovom modulu bude dobro definisana.

Prilikom analize problema identifikovani su sljedeći moduli:

- Rukovalac MPD dokumentom
- Rukovalac HTTP protokolom
- Rukovalac segmentima
- Modul za reprodukciju media sadržaja
- Adaptaciona logika

Na slici 3.1 prikazan je model rješenja na najvišem nivou apstrakcije. Pored glavnih modula na slici se takođe mogu vidjeti dva modula koji pasivno utiču na rješenje a to su poslužilac i zahtjevi korisnika. Klijent može od poslužioca zahtijevati segmente sa bajt opsegom samo ukoliko poslužilac podržava HTTP/1.1 protokol i uticaj poslužioca na rješenje

se tu i završava. U konkretnoj implementaciji je pretpostavljeno da poslužilac podržava HTTP/1.1 protokol.



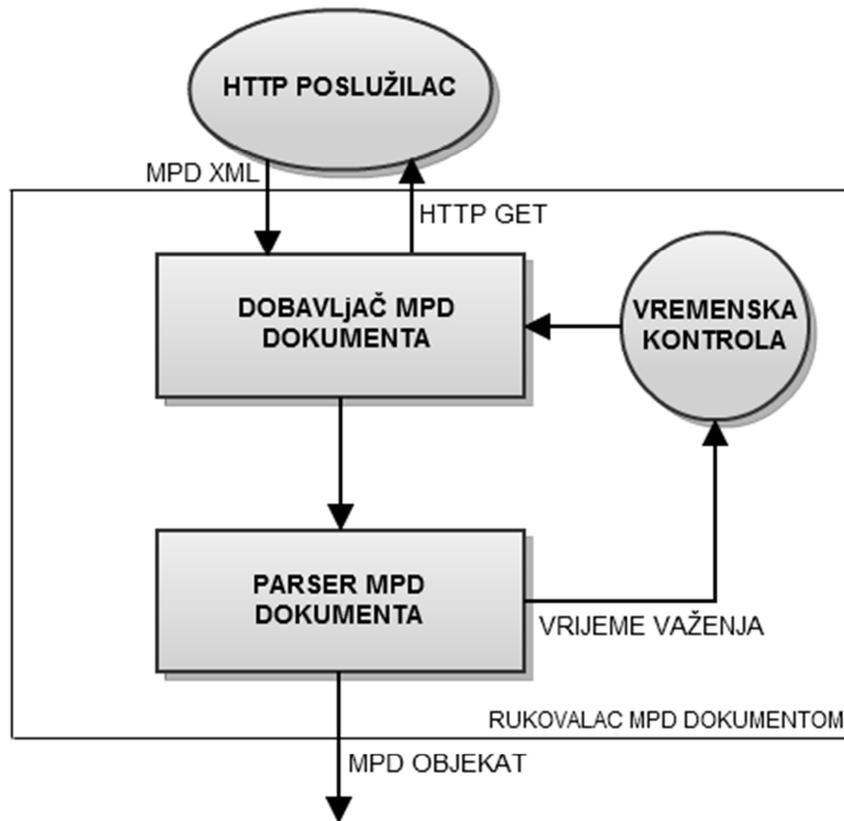
Slika 3.1 Model rješenja dinamičkog adaptivnog prenosa podataka

Zahtjevi korisnika predstavljaju spregu između korisnika i DASH klijenta. DASH klijent nastoji da ispuni zahtjeve korisnika, ali ako nije u mogućnosti da to uradi, bilo zbog nemogućnosti modula za reprodukciju medija sadržaja da isprati takve zahtjeve, bilo zbog nepostojanja odgovarajućeg sadržaja na strani poslužioca, DASH klijent će pregovarati sa korisnikom o alternativama ili će se potruditi da što približnije ostvari njegove zahtjeve.

3.2.1 Rukovalac MPD dokumentom

Rukovalac MPD dokumentom zadužen je za dobavljanje samog MPD dokumenta i njegovo parsiranje. Samim tim, ovaj modul može biti razložen na dva podmodula od kojih jedan podmodul predstavlja dobavljača MPD dokumenta a drugi podmodul predstavlja parser MPD dokumenta. Važno je naglasiti da rukovalac MPD dokumentom ne zavisi od MPEG-DASH profila koji se implementira tako da je moguće iskoristiti ovaj modul prilikom

implementiranja ostalih profila ili proširenja postojećeg rješenja bez ikakvih izmjena. Sprega ovog modula sa ostalim modulima je jednostavna i svodi se na to da je ulaz u modul XML MPD dokument a izlaz iz njega programski objekat parsiranog dokumenta koji modeluje njegov sadržaj i hijerarhiju. Na slici 3.2 prikazan je rukovalac MPD dokumentom.



Slika 3.2 Rukovalac MPD dokumentom

3.2.1.1 Dobavljač MPD dokumenta

Dobavljač MPD dokumenta zadužen je da šalje zahtjeve za MPD dokumentom HTTP poslužiocu. Kada dobije odgovor od poslužioca MPD dokument se prosljeđuje parseru. Dobavljač MPD dokumenta reaguje na signal isteka vremenske kontrole koji označava prestanak važenja dokumenta i potrebu za dobavljanjem ažurne verzije.

Ulaz u ovaj podmodul predstavljaju MPD dokument (odgovor HTTP poslužioca) i signal vremenske kontrole. Izlaz iz ovog podmodula je dobavljeni MPD dokument i HTTP zahtjev za dokumentom koji se šalje HTTP poslužiocu.

3.2.1.2 Parser MPD dokumenta

Ovaj podmodul je zadužen za parsiranje dobavljenog MPD dokumenta. Parsiranje se vrši na osnovu XML šeme MPD dokumenta koja je data standardom i nije zavisna od profila koji se implementira. Veoma važno je da parser bude efikasan po pitanju brzine kako ne bi predstavljao usko grlo i kako bi se smanjilo vrijeme od korisnikovog zahtjeva za reprodukcijom do početka reprodukcije medija sadržaja.

Rezultat parsiranja MPD dokumenta je programski objekat koji vijerno odslikava hijerarhiju dokumenta i sadrži sve informacije koje su u dokumentu navedene, a koje su od presudnog značaja za funkcionisanje dinamičkog adaptivnog prenosa podataka. Ovaj objekat predstavlja jedinu spregu između rukovaoca MPD dokumentom i ostala 3 glavna modula.

Ukoliko MPD dokument nije statički, parser ima obavezu da po saznavanju perioda osvježavanja sadržaja MPD dokumenta pokrene vremensku kontrolu čiji će istek označiti dobavljaču MPD dokumenta da je potrebno poslati zahtjev za ažurnom verzijom dokumenta.

3.2.2 Rukovalac HTTP protokolom

Funkcionalnost ovog modula se ogleda u efikasnom dobavljanju segmenata od HTTP poslužioca. Sprega ovog modula je jednostavna i svodi se na to da mu je potrebno obezbijediti URL i , ako je moguće, bajt opseg segmenta koji treba dobiti kako bi on mogao biti zatražen od HTTP poslužioca. Odgovor HTTP poslužioca koji sadrži traženi segment se smiješta na zadatu lokaciju u memoriji.

Dodatna funkcionalnost ovog segmenta se odnosi na mjerenje protoka u mreži i praćenje popunjenosti bafera, što je neophodno kako bi se mogla implementirati odgovarajuća adaptaciona logika.

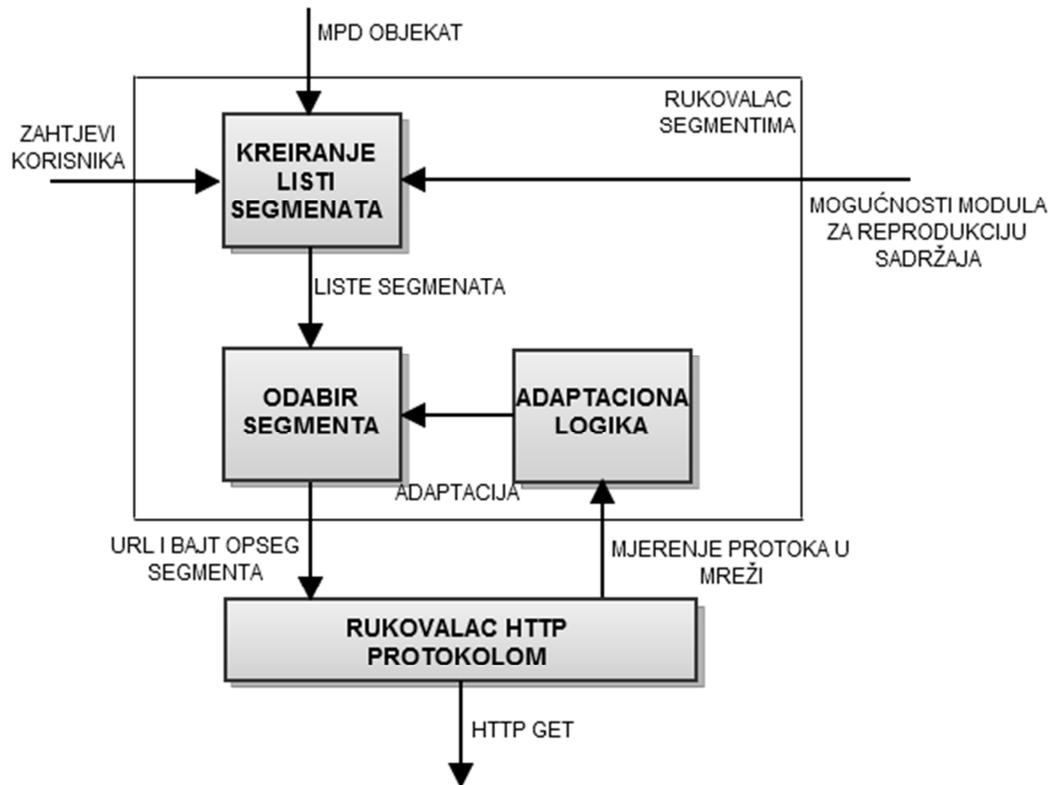
Ovaj modul takođe ne zavisi od profila koji se implementira tako da je moguće iskoristiti ga prilikom implementacije drugih profila ili proširenja postojećeg bez ikakvih izmjena.

3.2.3 Rukovalac segmentima

Rukovalac segmentima je centralni modul u okviru ovog rješenja. Njegova funkcionalnost se ogleda u stvaranju listi segmenata, odabiru segmenta i formiranju zahtjeva

koji se prosljeđuju rukovaocu HTTP protokolom kako bi segmenti bili dobavljeni. Ovaj modul je jedini koji zavisi od profila koji se implementira mada je moguće neke njegove dijelove iskoristiti prilikom implementacije ostalih profila.

U okviru ovog modula nalazi se i modul zadužen za implementaciju adaptacione logike. Na slici 3.3 prikazan je modul rukovaoca segmentima.



Slika 3.3 Rukovalac segmentima

3.2.3.1 Kreiranje listi segmenata

Zadatak ovog podmodula je kreiranje listi dostupnih segmenata na osnovu informacija dobijenih parsiranjem MPD dokumenta koje se nalaze u MPD programskom objektu. Postoje dva različita načina za opis i generisanje liste segmenata, na osnovu elementa liste (SegmentList element) i na osnovu elementa šablona (SegmentTemplate element) u MPD dokumentu. Trivijalni slučaj kada postoji samo jedan medija segment u okviru reprezentacije neće biti razmatran.

Ukoliko reprezentacija sadrži ili nasljeđuje element šablona segmenta onda se korišćenjem ovog šablona stvaraju liste. Standardnom MPEG-DASH detaljno je opisana procedura po kojoj se kreiraju liste segmenata na osnovu šablona segmenta.

Ukoliko reprezentacija sadrži ili nasljeđuje element liste segmenata onda je procedura kreiranja liste segmenata dosta pojednostavljena. U standardu MPEG-DASH takođe postoji dataljan opis ove procedure.

Na to koje će reprezentacije ovaj podmodul uzimati u obzir prilikom kreiranja listi segmenata utiču i zahtjevi korisnika kao i mogućnosti modula za reprodukciju medija sadržaja.

3.2.3.2 Odabir segmenta

Ovaj podmodul ima zadatak da na osnovu rezultata adaptacionog algoritma odabere odgovarajući segment iz liste segmenata. Nakon toga on šalje relevantne informacije o segmentu rukovaocu HTTP protokolom kako bi se dobio traženi segment. U te relevantne informacije spadaju URL i bajt opseg segmenta i mjesto u memoriji gdje je potrebno da se dobavljeni segment smjesti.

Prilikom odabira segmenta potrebno je obratiti pažnju na period vremena u kome je taj segment dostupan, jer će traženje nedostupnog segmenta od poslužitelja rezultovati greškom.

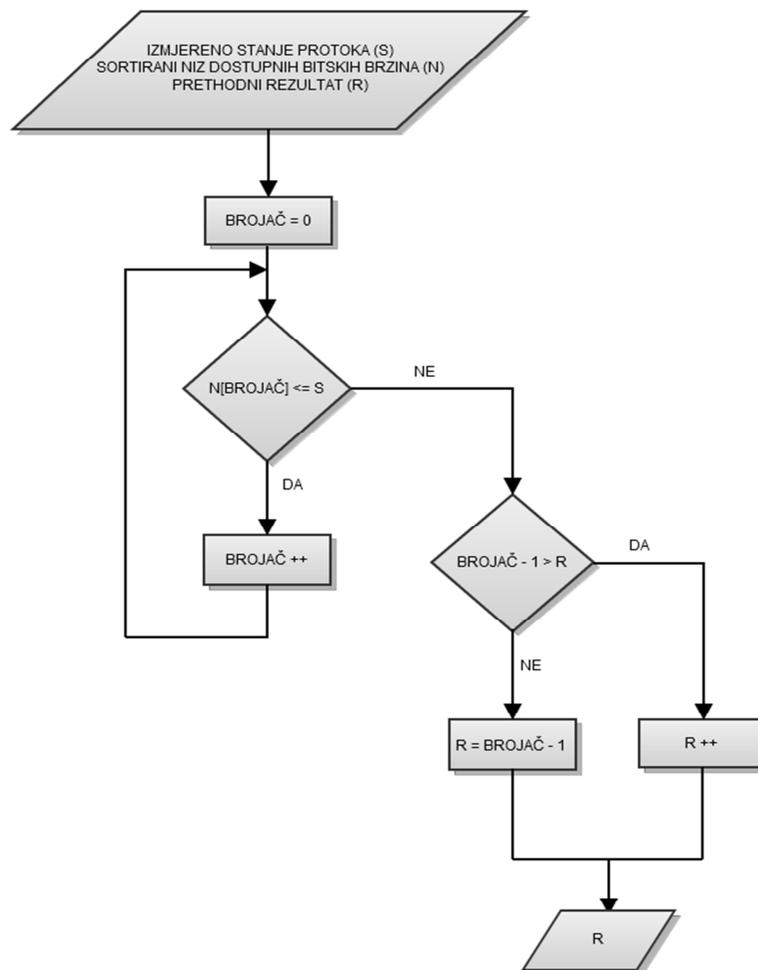
3.2.3.3 Adaptaciona logika

Adaptaciona logika je dio klijenta koji nije specificiran MPEG-DASH standardom. Od ovog modula najviše zavisi kakav će biti kvalitet usluge koja se pruža korisniku.

Napredni adaptacioni algoritmi su uglavnom kompleksni i predstavljaju zasebnu oblast istraživanja koja izlazi van okvira ovog rada. Ipak, potrebno je pažnju posvetiti sprezi prema ovom modulu kako bi se pojednostavilo ugrađivanje bilo kog adaptacionog algoritma u postojeće rješenje. Ulaz u ovaj modul je mjerenje izvršeno od strane rukovaoca HTTP protokolom koje se odnosi na stanje protoka u mreži, i popunjenost memorije za privremeno čuvanje (*engl. buffer*) podataka. Cilj je da se odabir bitske brzine sljedećeg segmenta koga treba dobiti prilagođava stanju protoka u mreži i stanju memorije za privremeno čuvanje podataka a da se pri tom korisniku pruži što kvalitetniji medija sadržaj.

Rezultat adaptacionog algoritma je bitska brzina sljedećeg segmenta koji treba dobiti. Ova bitska brzina se prosljeđuje podmodulu za odabir segmenta koji na osnovu nje odabira iz koje liste segmenata će uzeti sljedeći segment.

Adaptacioni algoritam korišćen u ovom rješenju dat je na slici 3.4. Logika algoritma je jednostavna i svodi se na to da se pronađe najveća dostupna bitska brzina koja je manja od protoka u mreži. Ukoliko je ta brzina veća od bitske brzine trenutno odabrane reprezentacije onda se vrši preključivanje na prvu veću dostupnu bitsku brzinu. Ukoliko brzina nije veća onda se vrši preključivanje na najveću dostupnu bitsku brzinu koja je manja od protoka u mreži. Cilj algoritma je da se povećavanje bitske brzine radi postepeno bez obzira da li se dogodilo naglo povećanje protoka u mreži, a da smanjivanje bitske brzine prati naglo smanjivanje protoka u mreži. Ovim se postiže kontinualna reprodukcija medija sadržaja i u uslovima kada se stanje protoka u mreži naglo mijenja, ali na uštrb nešto sporijeg prelaska na bolji kvalitet medija sadržaja ukoliko se protok u mreži naglo poveća.



Slika 3.4 Algoritam adaptacione logike

3.2.4 Modul za reprodukciju medija sadržaja

Ovaj modul je specifičan po tome što jedini zavisi od fizičke arhitekture. Sastoji se iz dva podmodula, dekodera i podmodula za prikaz sadržaja. U zavisnosti od proizvođača uređaja ovaj modul je implementiran na različite načine zbog potreba da se neki dijelovi implementiraju u samoj fizičkoj arhitekturi kako bi se povećala efikasnost uređaja. Zbog toga je veoma važno da sprega ka ovom modulu bude što bolje definisana i što je moguće jednostavnija kako bi se obezbijedila što lakša prenosivost rješenja na uređaje različitih proizvođača.

Ovim rješenjem je predviđeno da jedina sprega sa modulom za reprodukciju medija sadržaja bude specijalno izdvojeni segment u memoriji iz koga će ovaj modul cirkularno čitati medija podatke i reprodukovati ih. Zbog sinhronizacije čitanja i upisa u ovu memoriju spregu je potrebno proširiti sa još dva signala. Ukoliko je memorija prazna modul za reprodukciju treba da čeka na signal koji označava da u memoriji ima podataka koje treba reprodukovati kako ne bi došlo do čitanja iz praznih memorijskih lokacija. Ukoliko je memorija popunjena rukovalac HTTP protokolom treba da čeka signal kojim se označava da ima dovoljno prostora u memoriji za upis dobavljenog segmenta kako ne bi došlo do prepisivanja nereprodukovanih medija podataka. Podmodul za prikaz je usko povezan sa dekomerom i njegov zadatak je da na zvučnike reprodukuje dekodovani zvuk ili prikaže dekodovanu sliku na ekran.

4. Programsko rešenje

DASH klijent je realizovan sa dvije glavne niti. Prva nit je zadužena za dobavljanje segmenata od poslužioca a druga nit je zadužena za dekodovanje i reprodukciju medija sadržaja. Prije pokretanja ovih niti se izvrši dobavljanje MPD dokumenta i njegovo parsiranje. U ovom rješenju je pretpostavljeno da je MPD dokument statičan i zbog toga se on dobavlja i parsira samo jednom. Ukoliko bi MPD dokument trebalo periodično ažurirati onda bi to bilo urađeno u trećoj niti koja bi se periodično izvršavala.

Sinhronizacija između glavnih niti je obavljena preko kružnog bafera u koji nit za dobavljanje segmenata upisuje dobavljene podatke, a iz kog nit za reprodukciju čita podatke i reprodukuje ih. U okviru niti za dobavljanje segmenata se izvršavaju rukovalac segmentima i rukovalac HTTP protokolom, a u okviru niti za reprodukciju se izvršava modul za reprodukciju medija sadržaja.

U daljem tekstu će biti naveden način implementacije samo glavnih modula uz objašnjenje rada samo ključnih funkcija.

4.1 Rukovalac MPD dokumentom

Početak izvršavanja DASH klijenta vezan je za dobavljanje MPD dokumenta i njegovo parsiranje. Način dobavljanja dokumenta nije specificiran MPEG-DASH standardom. U ovom rješenju MPD dokument se dobavlja putem HTTP protokola HTTP GET zahtjevom. Dobavljeni MPD dokument se čuva u datoteci sa ekstenzijom .mpd. Ova datoteka predstavlja jedinu spregu između podmodula za dobavljanje MPD dokumenta i MPD parsera. Prototip funkcije za dobavljanje MPD dokumenta:

```
intgetMPD (char* url, char* directory);
```

Parametri:

- *url* – URL pomoću kog se formira HTTP GET zahtjev
- *directory* – direktorijum u koji je će biti smještena .mpd datoteka sa sadržajem MPD dokumenta (ime .mpd datoteke se formira na osnovu URL-a)

Povratna vrijednost:

- -1 – dobavljanje MPD dokumenta je neuspješno
- 0 – dobavljanje MPD dokumenta je uspješno

Parser MPD dokumenta je klasični XML parser koji informacije iz dokumenta pretvara u programski objekat koji u potpunosti odslikava hijerarhiju dokumenta. Za parsiranje XML dokumenata postoje već razvijeni algoritmi koji kreiraju objektni model dokumenta (DOM) u obliku stabla. U ovom rješenju korišćena je biblioteka RapidXml za kreiranje objektnog modela dokumenta a zatim je od tog objektnog modela kreiran programski objekat MPD dokumenta. RapidXml je veoma brz i stabilan XML parser napisan u jeziku C++. Optimizovan je za korišćenje u namjenskim uređajima i aplikacijama gdje iskorišćenje memorije i procesna moć imaju primarni značaj. RapidXml je pod dvostrukom licencom, MIT i Boost Software, i njegov izvorni kod je dostupan besplatno. Još jedan od važnih razloga zbog koga je baš ovaj parser izabran da bude ugrađen u ovo rješenje jeste njegova prenosivost (*engl. portability*). Funkcija za parsiranje MPD dokumenta:

```
intdomParser (char* MPDfile, MPD* mpd);
```

Parametri:

- *MPDfile* – putanja do MPD dokumenta
- *mpd* – objekat klase MPD koji će po završetku parsiranja sadržati sve informacije date u MPD dokumentu i u potpunosti odslikavati hijerarhiju dokumenta

Povratna vrijednost:

- -1 – parsiranje je neuspješno
- 0 – parsiranje je uspješno

Ponovno dobavljanje MPD dokumenta zavisi od stanja promjenljive koja signalizira istek vremenske kontrole. Vremenska kontrola je realizovana kao nit koja se na samom početku izvršavanja suspenduje onoliko vremena koliki je period osvježavanja MPD dokumenta. Kada se

nit probudi ona postavi promjenljivu koja označava da je vremenska kontrola istekla i završava se. Ovo je primitivan način realizacije vremenske kontrole ali sasvim zadovoljavajući za ovaj slučaj jer nije potrebno da preciznost vremenske kontrole bude velika.

4.2 Rukovalac HTTP protokolom

Rukovalac HTTP protokolom predstavlja klasičnog HTTP klijenta. Za njegovu realizaciju korišćena je libcurl biblioteka. Libcurl je besplatna i laka za korišćene biblioteka koja se koristi u klijentskim aplikacijama za prenos datoteka. Biblioteka je bezbjedna za korišćenje u okruženju niti, kompatibilana sa IPv6, prenosiva, brza, temeljno dokumentovana i široko korišćena. Biblioteka podržava i mjerenje protoka u mreži u toku dobavljanja sadržaja od poslužioca tako da u potpunosti odgovara zahtjevima koji su postavljeni pred rukovaoca HTTP protokolom. Funkcija za dobavljanje segmenta:

```
inthttp_get (RingBuffer* ringBuffer, char* url, char* byteRange, double*
downSpeed);
```

Parametri:

- *ringBuffer* – kružni bafer u koji se upisuju dobavljeni podaci
- *url* – URL segmenta
- *byteRange* – bajt opseg segmenta, ukoliko je poznat
- *downSpeed* – izlazni parametar u koji se upisuje prosječna brzina kojom je dobavljen segment (direktno ukazuje na stanje protoka u mreži)

Povratna vrijednost:

- -1 – dobavljanje segmenta je neuspješno
- 0 – dobavljanje segmenta je uspješno

Biblioteka podrazumijeva da se dobavljeni sadržaj upisuje u datoteku. Kako bi se omogućilo da se dobavljeni sadržaj upisuje u kružni bafer potrebno je napisati funkciju koja će to da radi i naglasiti libcurl biblioteci da je koristi umjesto podrazumijevane funkcije. Takva funkcija mora da ispoštuje sljedeći prototip:

```
size_t write_data(void* ptr, size_t size, size_t nmemb, void* userp);
```

Parametri:

- *ptr* – podaci koji su dobavljeni
- *size* – širina podatka u bajtima
- *nmemb* – broj podataka
- *userp* – korisnički podaci, u ovom slučaju predstavljaju adresu u memoriji gdje treba upisati dobavljene podatke (kružni bafer)

Povratna vrijednost:

- dužina podataka koja je upisana u memoriju, ukoliko je različita od dužine dobavljenih podataka biblioteka vraća grešku

U okviru ove funkcije je obezbijeđena sinhronizacija sa modulom za reprodukciju medija sadržaja koji ima komplementarnu funkciju za čitanje sadržaja iz kružnog bafera. Ukoliko u baferu nema dovoljno mjesta da se upišu dobavljeni podaci veličine $size \times nmemb$ onda se ova funkcija blokira (samim tim i HTTP klijent koji dobavlja segmente) i čeka se da funkcija iz modula za reprodukciju medija sadržaja koja čita podatke iz kružnog bafera signalizira da je oslobođeno dovoljno mjesta da se dobavljeni podaci upišu. Nakon upisa dobavljenih podataka u kružni bafer signalizira se funkciji iz modula za reprodukciju da ima podataka za čitanje.

4.3 Modul za reprodukciju medija sadržaja

Ovaj modul je specifičan po tome što jedini zavisi od fizičke arhitekture. Način implementacije ovog modula nije specificiran MPEG-DASH standardom. Sastoji se iz dva podmodula, dekodera i podmodula za prikaz sadržaja. Različiti proizvođači uređaja ovaj modul implementiraju na različite načine zbog potrebe da se neki dijelovi implementiraju u samoj fizičkoj arhitekturi kako bi se povećala efikasnost uređaja. Zbog toga je sprega ka ovom modulu definisana na način da se obezbijedi što lakša prenosivost rješenja na uređaje različitih proizvođača.

Sprega sa ovim modulom se svodi na kružni bafer u koji se upisuju dobavljeni medija podaci. Zbog sinhronizacije čitanja i upisa u ovu memoriju sprega je proširena sa još dva signala. Ukoliko je memorija prazna modul za reprodukciju treba da čeka na signal koji označava da u memoriji ima podataka koje treba reprodukovati kako ne bi došlo do čitanja iz praznih memorijskih lokacija. Ukoliko je memorija popunjena rukovalac HTTP protokolom treba da čeka signal kojim se označava da ima dovoljno prostora u memoriji za upis dobavljenog segmenta kako ne bi došlo do prepisivanja nereprodukovanih medija podataka. Podmodul za

prikaz sadržaja je usko povezan sa sa dekomerom i svodi se na reprodukovanje dekodovanog zvuka na audio uređaje i dekodovane slike na ekran.

Za potrebe demonstracije valjanosti imlementacije MPEG-DASH standarda za dekodovanje medija sadržaja korišćena je ffmpeg biblioteka, a za reprodukciju dekodovanog sadržaja korišćena je SDL biblioteka.

Modul je realizovan sa dvije glavne niti. Nit za parsiranje čita podatke iz kružnog bafera parsira ih i prosljeđuje dalje niti za dekodovanje i prikaz sadržaja. U okviru funkcije za reprodukciju video sadržaja enkapsulirana je funkcionalnost dekodovanja i prikaza sadržaja. Unutarnje se pokreću dvije gore pomenute niti.

```
int play(RingBuffer* ringBuffer, VideoState* is);
```

Parametri:

- *ringBuffer* – kružni bafer u kome se nalaze podaci
- *is* – struktura sa stanjem modula za reprodukciju (parametar je neophodan zbog rekurzivnosti funkcije)

Povratna vrijednost:

- -1 – greška u toku reprodukcije
- 0 – reprodukcija završena uspješno

Biblioteka podrazumijeva da se video i audio podaci čitaju iz datoteke tako da je bilo potrebno napisati funkciju koja će te podatke čitati iz kržnog bafera i naglasiti ffmpeg biblioteci da za čitanje medija podataka koristi tu funkciju umjesto podrazumijevane. Takva funkcija mora da ispoštuje sljedeći prototip:

```
int ReadFunc(void* opaque, uint8_t* buffer, int buf_size);
```

Parametri:

- *opaque* – korisnički podaci, u ovom slučaju kružni bafer iz kog treba čitati media podatke
- *buffer* – interni bafer u koji se smiještaju pročitani podaci koji biblioteka dalje koristi
- *buf_size* – veličina podataka koje treba pročitati

Povratna vrijednost:

- veličina stvarno pročitanih podataka

Ffmpeg biblioteka nije u stanju da se izbori sa dekodovanjem video sadržaja kome se dinamički mijenjaju dimenzije, tačnije, dekodier nije u stanju da dinamički mijenja svoj kontekst što je potrebno prilikom dekodovanja sadržaja kome se dinamički mijenjaju dimenzije. Slučaj dinamičkog mijenjanja dimenzija video sadržaja je sasvim uobičajen kada je dinamički adaptivan prenos video podataka u pitanju i zbog toga je bilo potrebno obezbijediti podršku za to.

Problem je riješen tako što se u trenucima kada je potrebno izvršiti preključivanje na reprezentaciju koja ima različite dimenzije od trenutno odabrane reprezentacije šalju signali za završavanje glavnih niti čime se praktično završava i funkcija za reprodukciju ali se prije izlaza iz funkcije vrši rekurzija. Kada se funkcija ponovo pozove izvrši se i ponovna inicijalizacija konteksta dekodera ali sa video sadržajem koji ima druge dimenzije. Rekurzija je izabrana kao rješenje koje će pružiti najbolje performanse jer je veoma važno da se preključivanje desi što je brže moguće da bi se obezbijedila kontinualnost u reprodukciji. Da se ne bi svaki put inicijalizovala struktura modula za reprodukciju ona se inicijalizuje pri prvom pozivu i u ostalim pozivima prosljeđuje kao parametar rekurzije.

Kako ne bi došlo do beskonačne rekurzije, prije svakog ponovnog poziva funkcije se provjerava da li je potrebno preključiti se na drugu reprezentaciju. Ako jeste onda se funkcija poziva ponovo, ako nije, funkcija se završava.

4.4 Rukovalac segmentima

Rukovalac segmentima se sastoji od tri podmodula od kojih je svaki enkapsuliran u zasebnu funkciju. Funkcija kojom je realizovan podmodul za kreiranje listi segmenata:

```
int createSegmentLists(MPD* mpd, SegmentManager* segmentManager, UsrPref* usr,
PlayerCapabilities* playerCap);
```

Parametri:

- *mpd* – objekat koji sadrži informacije iz MPD dokumenta i modeluje njegovu hijerarhiju
- *segmentManager* – objekat koji sadrži liste segmenata koje je potrebno popuniti
- *usr* – zahtjevi korisnika koji su vezani za karakteristike sadržaja i uglavnom se odnose na željeni kvalitet sadržaja

- *playerCap* – restrikcije koje uvodi modul za reprodukciju medija sadržaja, uglavnom se odnose na bitske brzine i kodeke koje podržava modul za reprodukciju

Povratna vrijednost:

- -1 – neuspješno kreiranje listi segmenata
- 0 – uspješno kreiranje listi segmenata

Funkcija na samom početku iz MPD objekta izdvaja one reprezentacije koje se poklapaju sa zahtjevima korisnika i koje ne izlaze van mogućnosti i ograničenja modula za reprodukciju medija sadržaja. Ovim postupkom se dobije skup reprezentacija koje dolaze u obzir za dinamički adaptivan prenos podataka. Za svaku od ovih reprezentacija se kreira lista segmenata. U okviru ovog rješenja nije predviđeno da se zahtjevi korisnika i mogućnosti modula za reprodukciju dinamički mijenjaju, već je pretpostavljeno da su ove informacije statične i poznate na početku dinamičkog adaptivnog prenosa podataka.

Liste segmenata se kreiraju ili na osnovu elementa šablona segmenta ili na osnovu elementa liste segmenata u zavisnosti od toga koji od ovih elemenata je prisutan u dokumentu. Oba načina kreiranja listi segmenata su detaljno opisana u MPEG-DASH standardu i u ovom rješenju je u potpunosti ispraćen taj opis. Po završetku funkcije se u objektu *segmentManager* nalaze popunjene sve liste segmenata koje dolaze u obzir za dinamički adaptivan prenos podataka. Liste segmenata su takođe sortirane u niz po rastućim bitskim brzinama.

Funkcija kojom je realizovan podmodul za odabir segmenta:

```
int chooseSegment(SegmentManager* segmentManager, int adaptation, Segment*
segment, int* index);
```

Parametri:

- *segmentManager* – objekat koji sadrži niz kreiranih listi segmenata sortiran po rastućim bitskim brzinama
- *adaptation* – index u nizu listi segmenata koji predstavlja rezultat adaptacionog algoritma
- *segment* – odabrani segment
- *index* – indeks segmenta koji je potrebno dobiti

Povratna vrijednost:

- -1 – segment nije uspješno odabran
- 0 – segment je uspješno odabran

Na osnovu parametra *adaptation* se pronalazi odgovarajuća lista segmanata u nizu listi segmanata. Na osnovu parametra *index* se iz liste uzima odgovarajući segment i dodjeljuje izlaznom parametru *segment*. Indeks se inkrementira prilikom svakog poziva funkcije za odabir segmenta kako bi uvijek ukazivao na indeks sljedećeg segmenta koji je potrebno odabrati.

Funkcija za realizaciju adaptacione logike:

```
int adapt(unsigned int* n, int speed, vector<int> bandwidths, double
bufferState);
```

Parametri:

- *n* – redni broj trenutno odabrane reprezentacije u koji će biti upisan redni broj reprezentacije iz koje se uzima sljedeći segment (rezultat adaptacije)
- *speed* – prosječna brzina kojom je skinut prethodni segment
- *bandwidths* – niz bitskih brzina koje dolaze u obzir za dinamički adaptivan prenos podataka
- *bufferState* – stanje popunjenosti bafera izraženo u procentima, ne koristi se u implementaciji postojećeg adaptacionog algoritma ali je važan parametar prilikom implementacije naprednih algoritama

Povratna vrijednost:

- -1 – greška u adaptacionom algoritmu
- 0 – adaptacija je izvršena bez greške

Niz bitskih brzina je kreiran na osnovu niza listi segmenata i vrijednost na nekom rednom broju u nizu bitskih brzina odgovara bitskoj brzini reprezentacije čija lista segmenata se nalazi na istom rednom broju u nizu listi segmenata. Ovakvom strukturom je ubrzano pronalaženje liste segmenata odakle je potrebno preuzeti informacije o sljedećem segmentu koji treba dobiti tako što je svedeno na indeksiranje.

Funkcija implementira jednostavan algoritam adaptacije prikazan na slici 3.4. Značaj ovog podmodula se ogleda i u tome što je definisana jasna sprega kako bi se lako mogli implementirati napredniji algoritmi za adaptaciju.

5. Ispitivanje i verifikacija

Realizacija je ispitana mehanizmom *unit-test*, a korišćena je biblioteka CPPUNIT za ispitivanje. Napravljeni su posebni testni skupovi za svaki modula izuzetkom modula za reprodukciju, a svaki testni skup sadrži testove (testne slučajeve, *test cases*) raznih segmenata tog modula – funkcija i struktura, kao i ispitivanje njihovih međusobnih odnosa. Modul za reprodukciju nije ispitivan zato što je realizovan korišćenjem već postojećih ispitanih biblioteka.

Rješenje je robusno i stabilno. Ni u jednom trenutku ne dolazi do prekida reprodukcije medija sadržaja zbog ispražnjenosti bafera, čak i u uslovima velikih fluktuacija protoka u mreži. Iskorišćenost protoka je takođe na jako visokom nivou jer se skoro u svakom trenutku dobavlja sadržaj najveće ponuđene bitske brzine koja je manja od propusnog opsega u mreži. Izuzetak predstavlja slučaj naglog povećanja protoka u mreži jer se prelazak na veće bitske brzine dešava postepeno. Adaptaciona logika implementirana u ovom rješenju je jednostavna ali veoma pouzdana i pruža dobre performanse.

Dobra iskorišćenost memorije je svakako jedan od prioriteta kada se radi o programskoj podršci za namjenske uređaje. Sasvim je uobičajeno da aplikacije koje imaju veze sa obradom multimedijalnog sadržaja troše puno memorije, pa je takav slučaj i sa testnom aplikacijom koja demonstrira rad dinamičkog adaptivnog prenosa podataka. Međutim, kada je riječ o samoj implementaciji standarda koja ne uključuje modul za reprodukciju sadržaja, potrošnja memorije je minimalna. Ono što je takođe veoma važno naglasiti jeste da ne postoje kritične tačke u rješenju koje zahtijevaju veliku procesnu moć, a to je posljedica dobre struktuiranosti rješenja kao i korišćenja velikog dijela već postojećih biblioteka koje su optimizovane za namjenske uređaje.

6. Zaključak

U prethodnim poglavljima opisan je jedan od načina implementacije ISO „Live“ profila MPEG-DASH standarda za dinamički adaptivan prenos podataka preko HTTP protokola. Posebna pažnja posvećena je pronalaženju optimalnog načina modularizacije s ciljem da se dobije razumljivo i logički korektno rješenje. Značaj ovog rada ogleda se u tome što pripada oblasti koja je u posljednje vrijeme veoma aktuelna, a to je dinamički adaptivan prenos podataka.

Veliki dio rješenja koji ne zavisi od profila koji je implementiran može biti ponovo iskorišćen prilikom implementacije ostalih profila MPEG-DASH standarda. Modul za reprodukciju medija sadržaja koji jedini zavisi od fizičke arhitekture ima jasno definisanu spregu čime je omogućena lakša prenosivost rješenja na različite platforme.

Obezbijeđena je podrška za efikasno preključivanje između reprezentacija različitih bitskih brzina i implementirana je jednostavna adaptaciona logika koja obezbjeđuje kontinualnu reprodukciju medija sadržaja.

Ostavljen je prostor za unapređenje rješenja integracijom naprednih adaptacionih algoritama. Još jedno od mogućih unapređenja ovog rješenja je obezbjeđivanje podrške za dinamičko mijenjanje zahtjeva klijenta i mogućnosti modula za reprodukciju medija sadržaja. Jedan od budućih ciljeva vezanih za ovo rješenje jeste njegova integracija u postojeću realizaciju digitalnog TV prijemnika baziranog na Android operativnom sistemu [5][6].

7. Literatura

- [1] ISO/IEC CD 23001-6: Information technology – MPEG systems technologies – Part 6: Dynamic adaptive streaming over HTTP (DASH) (2011)
- [2] ISO/IEC 14496-12:2008/DAM 3, Information Technology—Coding Of Audio-Visual Objects—Part 12: ISO Base Media File Format—Amendment 3: DASH Support and RTP Reception Hint Track Processing, Jan. 2011.
- [3] T. Stockhammer, TS 26.247 Transparent End-to-End Packet-Switched Streaming Service (PSS), Progressive Download and Dynamic Adaptive Streaming over HTTP, 3GPP, June 2011.
- [4] R. Fielding, UC Irvine, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee, Hypertext Transfer Protocol – HTTP/1.1 IETF Network Working Group, Request for Comments: 2616, April 1999.
- [5] M. Vidakovic, N. Teslic, T. Maruna, and V. Mihic, “Android4TV: a Proposition for Integration of DTV in Android Devices”, IEEE 30th International Conference on Consumer Electronics (ICCE), Las Vegas, January 2012, pp. 441-442.
- [6] K. Lazic, M. Milosevic, M. Kovacev, and N. Smiljkovic, “One Implementation of adaptive streaming over HTTP on Android DTV platform”, IEEE 2nd International Conference on Consumer Electronics (ICCE), Berlin, September 2012, accepted.