Verifikacija EBU-TT-D subtitle podrške u HbbTV okruženju

Hrbak, Mario

Master's thesis / Diplomski rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:005224

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

Verifikacija EBU-TT-D subtitle podrške u HbbTV okruženju

Diplomski rad

Mario Hrbak

Osijek, 2016.

Sadržaj

1. Uvod ... 1
2. Teorijske osnove ... 2
2.1 HbbTV ... 2
2.1.1 Pregled HbbTV-a ... 3
2.3 HTTP ... 6
2.4 Sustav za testiranje ... 7
2.4.1 Testni Paket ... 8
2.4.2 Testno okruženje ... 9
2.5 Kreiranje testova ... 10
2.6 Test API i definicija Playout Ser-a ... 11
2.6.1 API komunikacija s testnim okruženjem ... 13
2.6.2 API interakcija s uređajem pod testom ... 15
2.6.3 API komunikacija s Playout okruženjem ... 16
3. EBU-TT ... 18
3.1 XML titlovi ... 20
3.1.1 tt:tt ... 20
3.1.2 tt:head ... 20
3.1.3 tt:styling ... 20
3.1.4 tt:layout ... 21
3.1.5 tt:body ... 21
3.1.6 tt:div ... 21
4. EBU_TT Intepreter ... 23
4.1 Popis funkcija ... 27
5. Testovi ... 31
5.1 Opis izrade testa ... 31
5.2 Urađeni testovi ... 32
5.3 T2 Unit ... 35
6. Zaključak ... 36
Literatura ... 37
Sažetak ... 39

Summary 39
Životopis 40
Prilozi 41

1. Uvod

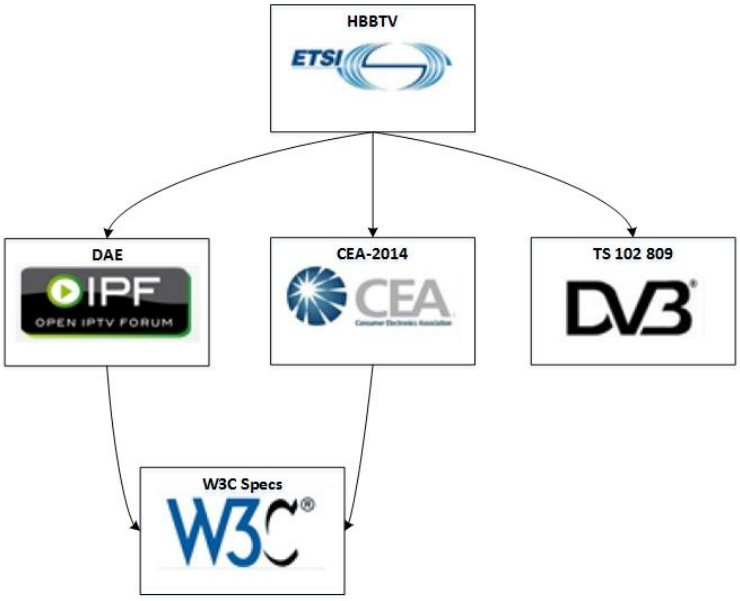
U radu će se kreirati i verificirati titlovi u HbbTV-u (eng. Hybrid Broadcast Broadband TV) prema EBU-TT-D normi. HbbTV je industrijski standard za proširenje standardnih mogućnosti digitalne televizije dodatnim interaktivnim aplikacijama zasnovan na web tehnologijama (html, css, JavaScript). Sadržaj je moguće prikazivati dobivanjem podataka iz prijenosnog toka, internetske veze ili drugih povezanih uređaja u okolici. Za komunikaciju se pretežno koristi HTTP (eng. HyperText Transfer Protocol) protokol zbog svoje rasprostranjenosti i isplativosti. Razvojem hibridne tehnologije javila se potraba za novim standardom za titlove u televiziji. Podrška za titlove u okviru HbbTv okruženja definirana je EBU-TT-D standardom, koji propisuje format dokumenata za distribuciju titlova zasnovan na XML-u (eng. EXtensible Markup Language). Ovakav format titlova se izvršava u web pretraživaču, te osim samog prikaza teksta omogućava stavljanje cijelih web stranica u titlove. U nastavku će biti dan pregled HbbTV standarda, s posebnim naglaskom na podršku za titlove. U nastavku će biti realiziran podskup testnih slučajeva kojima se verificira pravilna implementacija podrške za EBU-TT-D titlove korištenjem HbbTV test API-a (eng. Application programming interface) u skladu s pravilima testiranja koja definira HbbTV udruženje, te proširiti postojeća JavaScript biblioteku koja implementira osnovne EBU-TT-D funkcionalnosti sa svim funkcionalnostima koje se u testovima verificiraju.

2. Teorijske osnove

U ovom poglavlju će biti dan osvrt na teorijske osnove potrebne za razumijevanje rada kao što su HbbTV, EBU-TT-D standard, HTTP protokol, testno sučelje.

2.1 HbbTV

HbbTV je inicijativa kojom se razvio industrijski standard za razvoj hibridnih, interaktivnih DTV (eng. Digital television) aplikacija koje koriste emitirani prijenosni tok i vezu s internetom (*Broadcast* i *Broadband*). HbbTV je zasnovan na postojećim web standardima i tehnologijama. Na slici 2.1 je vidljiv odnos HbbTV-a s ostalim tehnologijama.

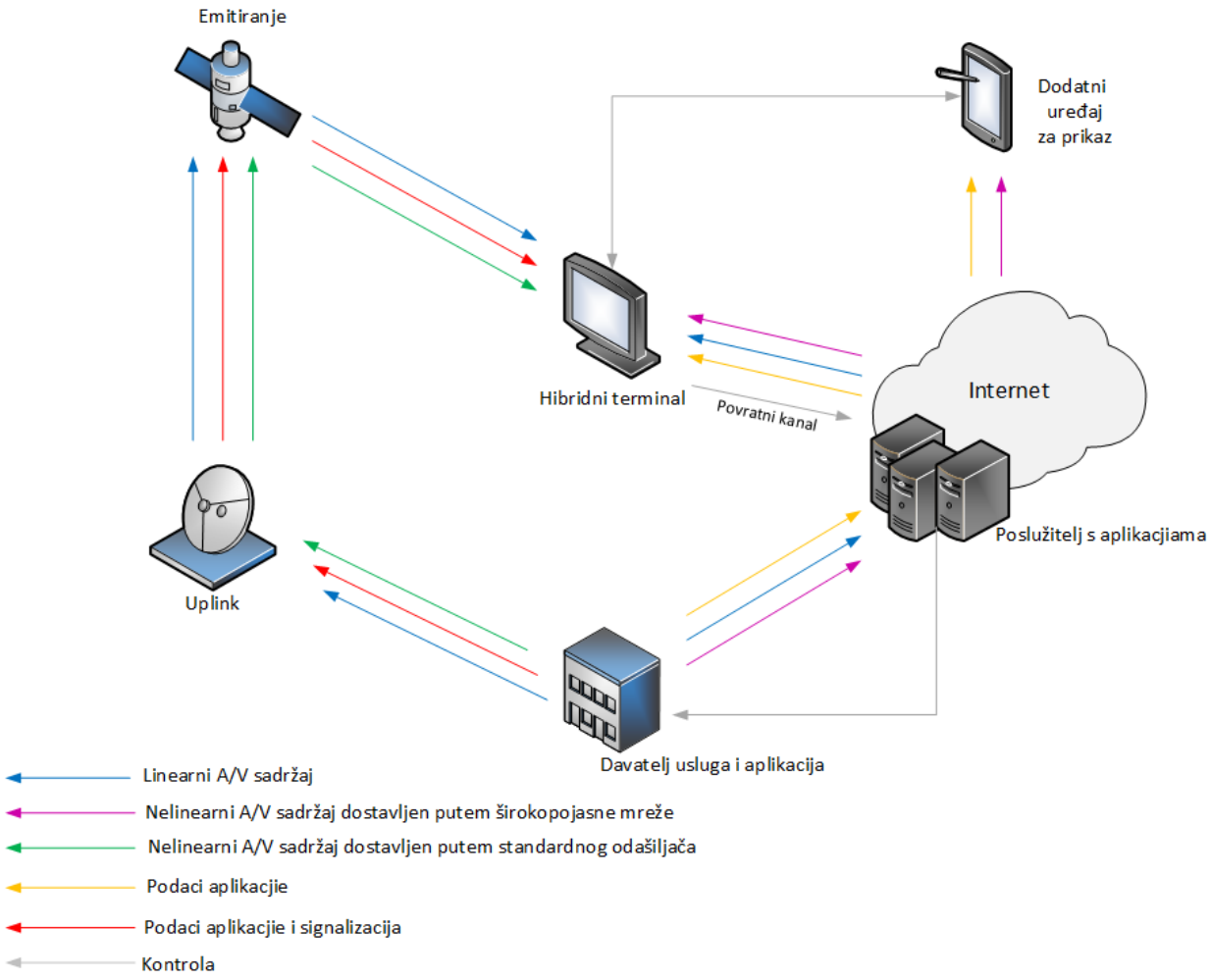


Slika 2.1 Odnos HbbTV-a s ostalim tehnologijama[1]

HbbTV za korištenje tipično traži DTV prijemnik s IP (eng. Internet Protocol) vezom i oslanja se na tehnologije vrlo poznate web programerima (HTML, CSS, JavaScript) koje mu daju prednost nad dosad korištenog MHEG-a (eng. Multimedia and Hypermedia Experts Group) koji ima svoj vlastiti jezik za opis interaktivnih TV usluga, što znači i ograničenu skupinu programera. Korištenje web tehnologija poboljšava brzinu razvoja novih aplikacija i lakše korištenje postojećih sadržaja. Usluge koje HbbTV obuhvaća su: poboljšani teletext, VOD (video na zahtjev), EPG (elektronski

programski vodič), društvene mreže, catch-up usluge, interaktivni oglasi, glasanja, igrice i multimedijske aplikacije.

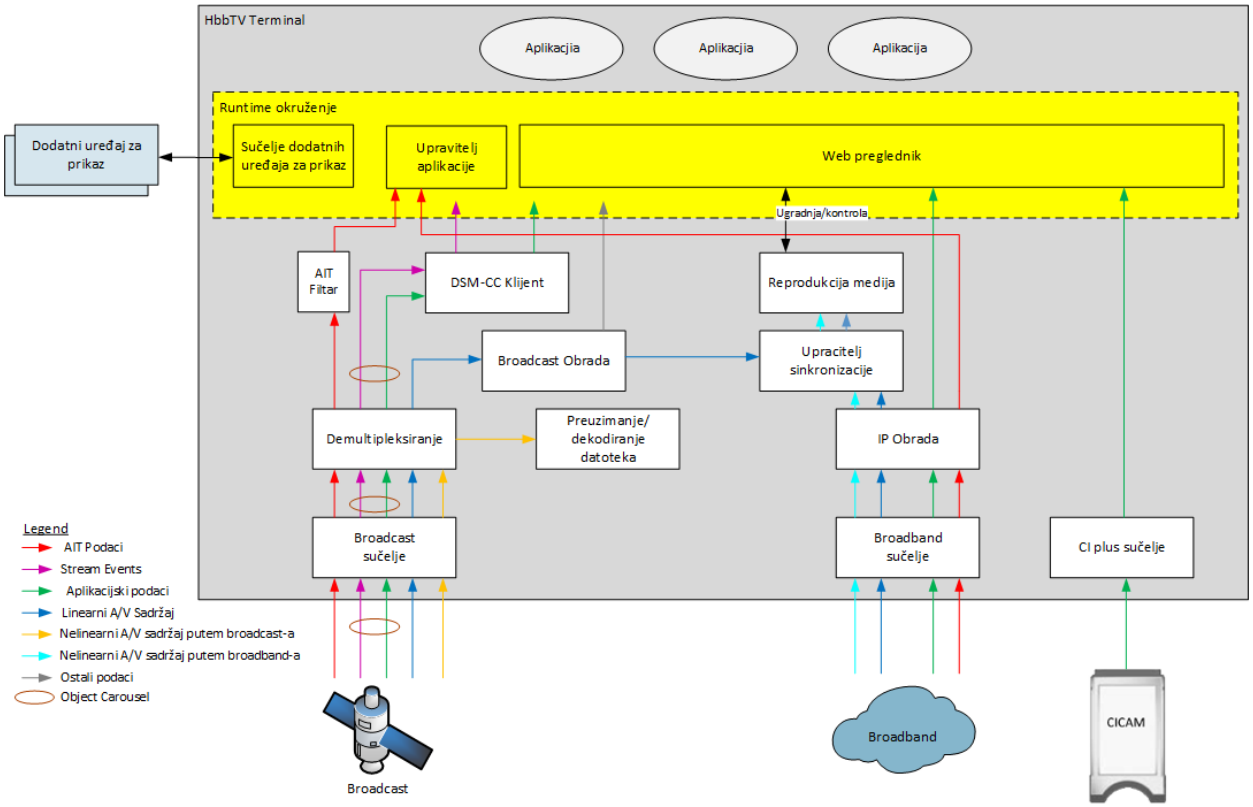
2.1.1 Pregled HbbTV-a



Slika 2.2 Prikaz sustava

Hibridna televizija omogućava istovremeni priključak na dvije mreže u paraleli vidljivo na slici 2.2. S jedne strane je priključena na odašiljačku DVB (eng. Digital Video Broadcasting) mrežu (*broadcast*), prijenos zemaljskim (DVB-T), kabelskim (DVB-C) ili satelitskim (DVB-S) putem. S druge strane je priključena na vezu s internetom (*broadband*). Putem DVB odašiljanja hibridni

terminal može primati standardni prijenosni tok te prikazivati televizijski program, multimedijske sadržaje koji se ne odašilju u stvarnom vremenu, te signalne informacije i podatke aplikacija. Čak i bez povezanosti na internet moguće je primati aplikacijske podatke vezane za prijenosni tok te također signalizirati aplikacijama događaje povezane s prijemnim tokom. U slučaju kada postoji povezanost s internetom, omogućuje se dvosmjerna veza između korisnika i davatelja usluge. Ova veza omogućuje primanje aplikacijskih podataka i multimedijskih sadržaja koji nisu u stvarnom vremenu, omogućuje se kontrola aplikacija i puštanje multimedijskog sadržaja po želji, te također može postojati mogućnost njegovog spremanja na neki memorijski uređaj. Druga stvar koja se omogućuje je povezivanje dodatnih uređaja u lokalnu mrežu, kao na primjer dodatnih HbbTV terminala ili nekih drugih pametnih telefona ili tableta.



Slika 2.3 Fukncionalne komponente hibridnog terminala

Putem *Broadcast* sučelja terminal prima AIT podatke, linearni A/V (audio/video) sadržaj, A/V sadržaj koji nije u stvarnom vremenu, podatke aplikacija i događaje prijenosnog toka. Posljednja dva toka podataka se prenose koristeći DSM-CC (eng. Digital storage media command and control)

karusel objekt. za prijenos sadržaja koji nije u stvarnom vremenu koristi se FDP (eng. File Download Protocol) protokol. Tako da su i DSM-CC klijent i FDP dekoder potrebni za obnavljanje podataka iz karusel objekta i FDP podatkovnog toka. Obnovljeni podaci se prosljeđuju radnom okruženju. Radno okruženje se može gledati kao vrlo apstraktna komponenta gdje se prezentiraju i izvršavaju interaktivne aplikacije. Pretraživač, menadžer aplikacija i sučelje dodatnog ekrana za prikaz čine ovo radno okruženja. Menadžer aplikacija procjenjuje AIT za kontrolu životnog ciklusa interaktivnih aplikacija. Pretraživač je odgovoran za predstavljanje i izvršavanje interaktivnih aplikacija.

Linearni A/V sadržaj se obrađuje na isti način kao i na standardnom ne-hibridnom DVB terminalu. Ovo je uključeno u funkcionalnu komponentu nazvanu *Broadcast Processing* koja uključuje sve DVB funkcionalnosti koje pruža standardni ne-hibridni DVB terminal. Dodatno se nekim informacijama i funkcijama iz *Broadcast Processing* komponente može pristupiti iz radnog okruženja (npr. informacija o listi kanala, EIT p/f, funkcije za pretragu frekvencije). One su uključene u "*other data*" na slici. Štoviše aplikacija može skalirati i ugraditi linearni A/V sadržaj u korisničkom sučelju dobivenom od aplikacije. Ove funkcionalnosti su pružene od strane *Media Playera*. Na slici 2.3 to uključuje sve funkcionalnosti povezane sa obradom A/V sadržaja.

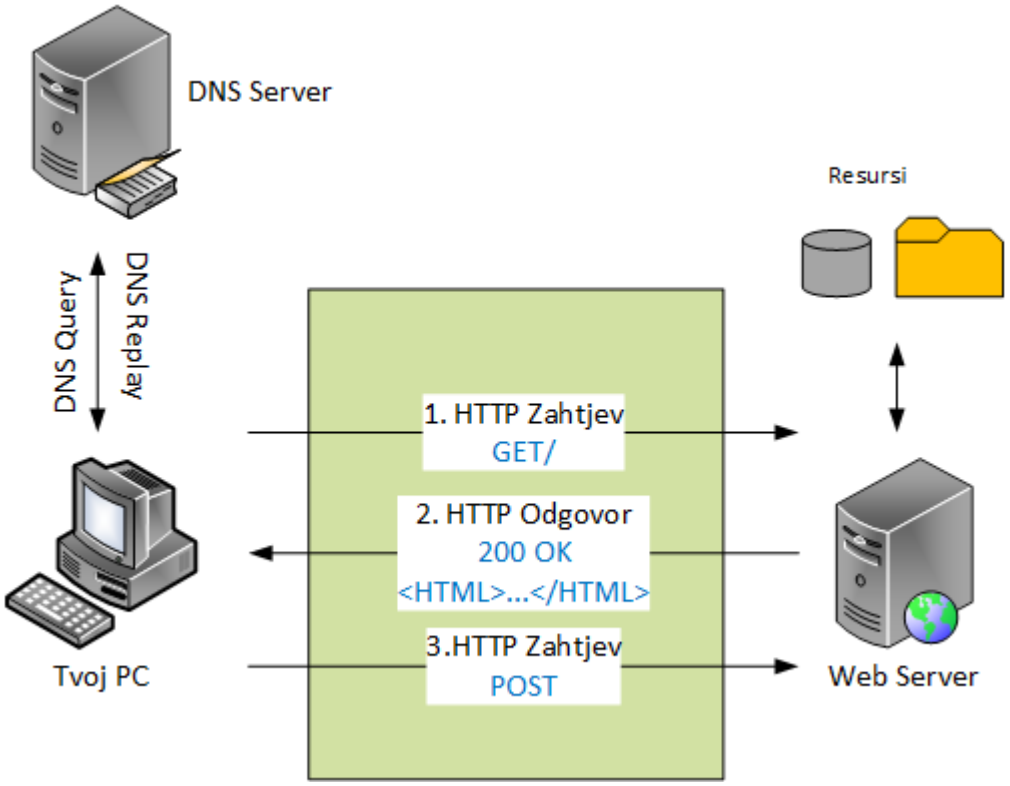
Pomoću *Broadb.and* sučelja hibridni terminal je povezan na internet. Ova veza pruža drugi put za traženja aplikacijskih podataka sa servera davatelja aplikacija. Također ova veza se koristi i za primanje A/V sadržaja (npr. sadržaj na zahtjev). Komponenta *Internet Protocol Processing* sadrži sve funkcionalnosti koje pruža terminal za rukovanje podacima sa interneta. Kroz ovu komponentu aplikacijski podaci se pružaju radnom okruženju. A/V sadržaj se prosljeđuje *Media Playeru* koji može biti kontroliran iz radnog okruženja te s time ugrađen u korisničko sučelje koje pruža aplikacija. U kombinaciji s *Media Playerom, Synchronization Manager* može sinkronizirati sadržaj dostavljen na hibridni terminal s *Broadband* sučelja i sadržaj dostavljen na hibridni terminal s ili *Broadband* sučelja ili *Broadcast* sučelja.

Companion Screen sučelje omogućava hibridnom terminalu da otkrije *Companion Screen* uređaje i druge hibridne terminale, te da može biti otkriven od strane drugih *Companion Screen* uređaja. Kroz njega, interaktivna aplikacija koja je pokrenuta u pretraživaču može zatražiti instaliranje ili pokretanje aplikacije na *Companion Screen* uređaju te aplikacija koja je na *Companion Screen* uređaju može zatražiti pretraživač da pokrene interaktivnu aplikaciju. Koristi se *WebSocket* server da omogući komunikaciju između interaktivne aplikacije na hibridnom terminalu i interaktivne

aplikacije na *Companion Screen* uređaju ili na drugom hibridnom terminalu. U kombinaciji *Companion Screen* sučelje i *Media Player* zajedno omogućuju sinkronizaciju sadržaja dostavljenog na hibridni terminal pomoću sučelja sa sadržajem dostavljenom na *Companion Screen* uređaj ili drugi hibridni terminal.

Pomoću CI plus sučelja, hibridni terminal zahtjeva aplikacijske podatke iz pomoćnog sustava datoteka ponuđenog od strane CICAM.

2.3 HTTP



Slika 2.4.HTTP komunikacija

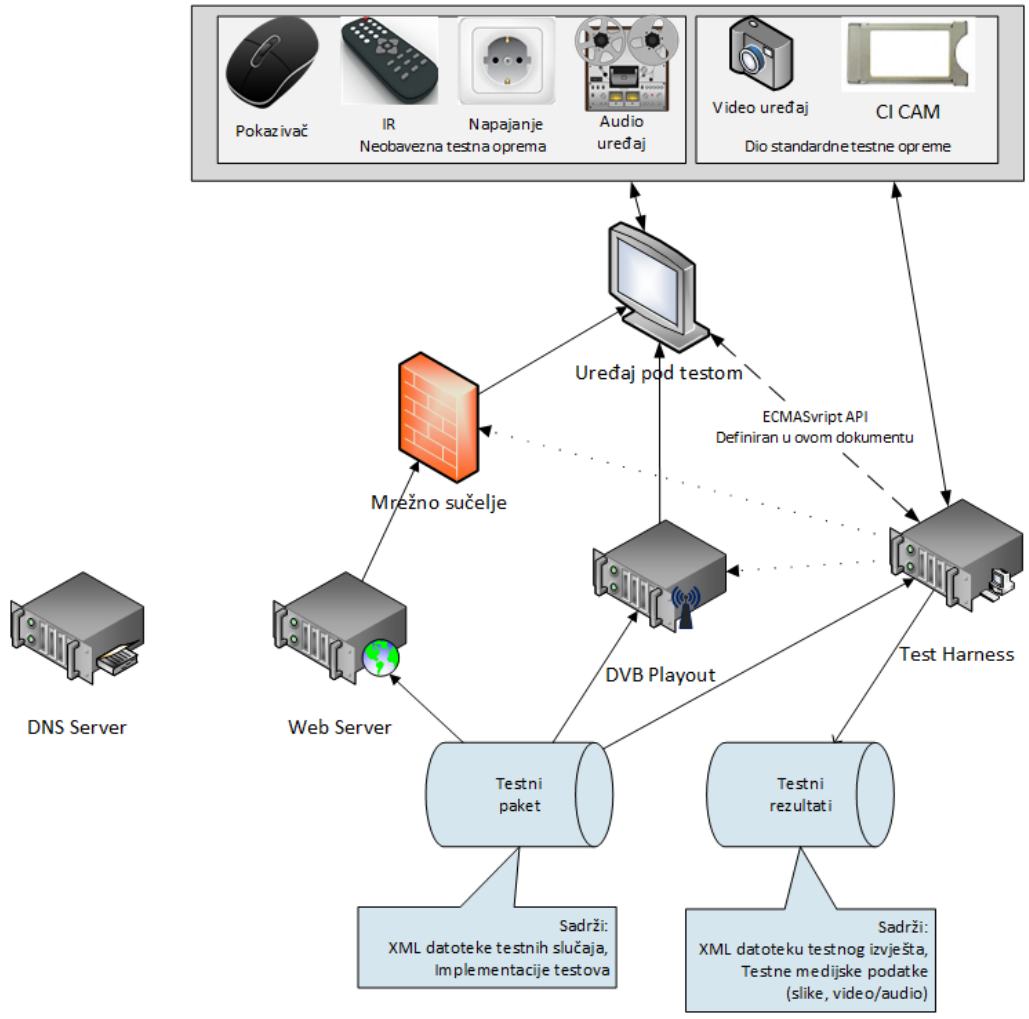
HTTP je najzastupljeniji protokol za prijenos putem web-a. HTTP je protokol aplikacijske razine kojim se vrši prijenos datoteka koje u sebi sadrže veze na druge dokumente. To je protokol za komunikaciju između klijenta i poslužitelja. Na slici 2.4 je prikazana komunikacija. Komunikacija se najčešće izvodi nakon što se uspostavi veza između klijenta i poslužitelja putem protokola kao što je TCP (eng. Transmission Control Protocol) ili UDP (eng. User Datagram Protocol). Server

konstantno osluškuje zahtjeve na određenom portu (najčešće port 80). Nakon toga klijent šalje zahtjev GET za uspostavu komunikacije i niz znakova koji određuju aspekte zahtjeva. Nakon GET mora se poslati još i niz zahtjeva koji određuju adresu klijenta. Na ovo klijent šalje odgovor i nakon toga tražene podatke ili poruku o grešci te se komunikacija zatvara. Zatvaranje komunikacije nakon prijenosa podataka je jedna od glavnih karakteristika HTTP protokola koja ga čini optimalnim za web. Nalazi se iznad TCP/IP protokola kojeg koristi za prijenos. HTTP protokolom definira se i forma komunikacije između klijenta i poslužitelja, tj. način postavljanja upita i odgovora i njihov format, kodiranje znakova karakterističnih za brojne jezike (*character set*), kodiranje sadržaja (*content coding*), kristup dokumentima za različite tipove protokola, kristup dokumentima uz provjeru identiteta (autorizacija i autentifikacija), spremanje dokumenata u privremenu memoriju (*caching*), Sigurnosne aspekte -osjetljive točke u komunikaciji između klijenta i poslužitelja (proces dostave podataka korisniku).

2.4 Sustav za testiranje

Sustav za pokretanje i implementiranje testova prikazan na slici 2.5. Sastoj se od:

1. Skupa odobrenih testnih slučaja, dobivenih i održavanih od strane HbbTV zajednice.
2. Testnog okruženja korištenog da se izvrše testni slučajevi i zabilježe rezultati testiranja koji se sastoji od:
    a. Uređaja pod testom (DUT).
    b. Test Harness-a (TH) koji služi kao testni operator te upravlja testnim slučajevima i rezultatima izvršavanja.
    c. Standardne testne opreme (STE).
    d. Opcionale testne opreme (OTE).
    e. RF i IP konekcije između uređaja pod testom i ostalih elemenata testnog okruženja.



Slika 2.5 Primjer testnog sistema

2.4.1 Testni Paket

Skup testova u HbbTV-u sastoji se od testnih slučajeva. Svaki test je jedinstveno označen i namijenjen da provjerava specifičan zahtjev iz HbbTV specifikacije. Testni slučaj se sastoji od :

- XML-a testnog slučaja - to je višenamjenski dokument koji sadrži :
  - Specifikacijske reference
  - Proceduru i očekivane rezultate
  - Povijest izmjena (razvojne verzije) testnih slučaja

- Konfiguracijskih datoteka koje su potrebne za izvršavanje testnog slučaja na Test Harnessu

- Testnih implementacijskih dokumenata (gdje više testnih slučaja može koristiti iste dokumente koji su dijeljeni)
- Informacije o licencama

Testni slučajevi u setu testova mogu imati jedan od dva statusa:

- Testni slučajevi odobreni od strane HbbTV testirajuće grupe formiraju dio zahtjeva za HbbTV usklađenost
- Testni slučajevi koji nisu odobreni od strane HbbTV testirajuće grupe. Oni su tada dani na korištenje programerima , te mogu biti odobreni ukoliko u budućnosti zadovolje HbbTV zahtjeve i razmotreni su od strane HbbTV testirajuće grupe.

2.4.2 Testno okruženje

Testno okruženje za izvršavanje HbbTV seta testova na DUT-u (eng. Device Under Test) sastoji se od dvije glavne komponente :

- Standardne testne opreme - to je skup svih alata koji su potrebni za pohranu, posluživanje, generiranje i puštanje testnih slučajeva na DUT-u. U to spadaju i web serveri i DVB *Playout Stream*. Komponente standardnog testnog okruženja nisu uključene s setom testova, ali možemo koristiti komercijalno dostupne alate.
- Test Harness - je sustav koji upravlja selekcijom i izvršenjem testnih slučajeva na DUT-u te s prikupljanjem rezultata i izvještaja testnih slučajeva.

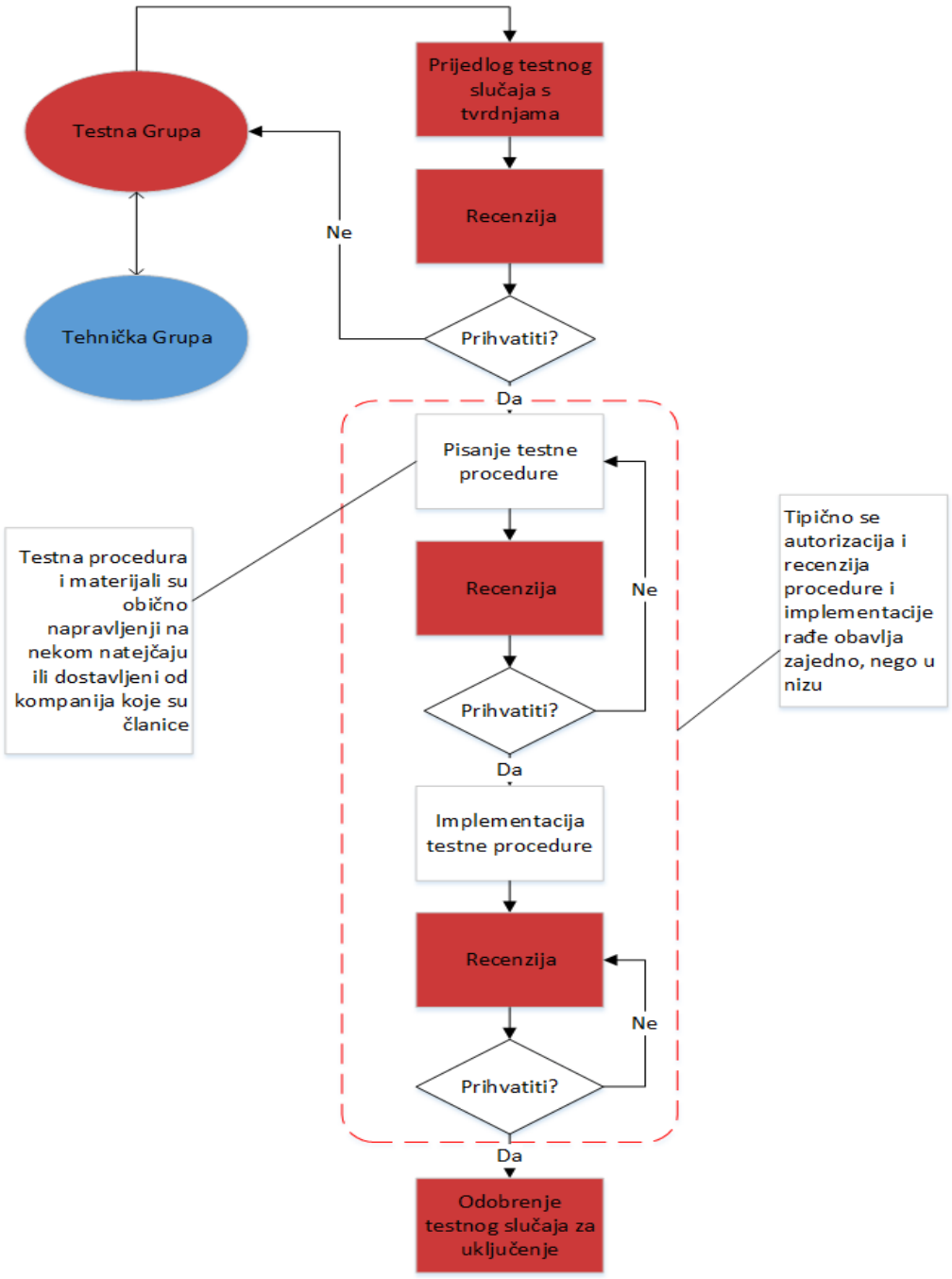
Test Harness:

1. Koristi informacije iz XML opisa HbbTV testnog slučaja i iz testnih materijala kako bi izvršio sve potrebne koraka prije izvršenja.
2. Započinje izvršenje testnog slučaja na DUT-u s time da pravi promjene u okruženju koje se temelje na vremenskim točkama koje su definirane u testnim materijalima i u odgovoru na API poziv iz testnog slučaja.
3. Sakuplja rezultate testnog slučaja koji se vrti na DUT-u

2.5 Kreiranje testova

HbbTV testni slučajevi su bazirani na proizvoljnim i obaveznim zahtjevima definiranim u HbbTV tehničkoj specifikaciji. Testni slučajevi su predloženi i vođeni od strane HbbTV testne grupe.

Proces definiranje, implementacije i prihvaćanja HbbTV testnih slučajeva sastoji se od koraka prikazanih u slici 2.6.



Slika 2.6 Dijagram toka nastanka testa

Test će proći ako su sljedeći kriteriji ostvareni:

- Svi obavezni preduvjeti testa su zadovoljeni
- Svi rezultati u test harnessu imaju vrijednost „*true*“ za parametar rezultat
- Svi pozivi za analiziranje API metoda daju rezultat „*true*“
- Svi pozivi testnih API metoda koji imaju interakciju s testnim okruženjem su uspješni
- Pozvana je „*endTest*“ API metoda

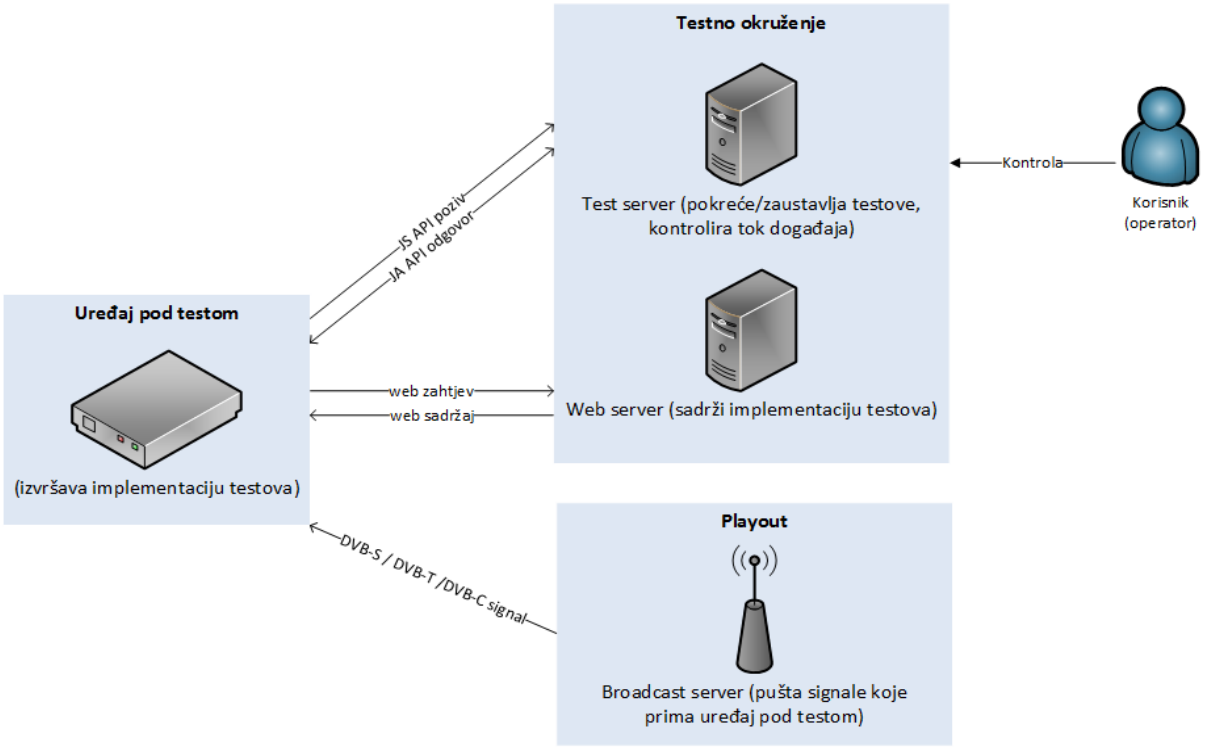
Ako bilo koji od prethodnih uvjeta nije ostvaren rezultat testa će biti pad. Rezultat se može promijeniti u prolaz ako se kriterij ostvari u kasnije u vremenu. Kako se rezultati i analiza rezultata ne može promijeniti, ako je u nekom trenutku u vremenu rezultat prolaz tada se rezultat ne može promijeniti u pad.

2.6 Test API i definicija Playout Ser-a

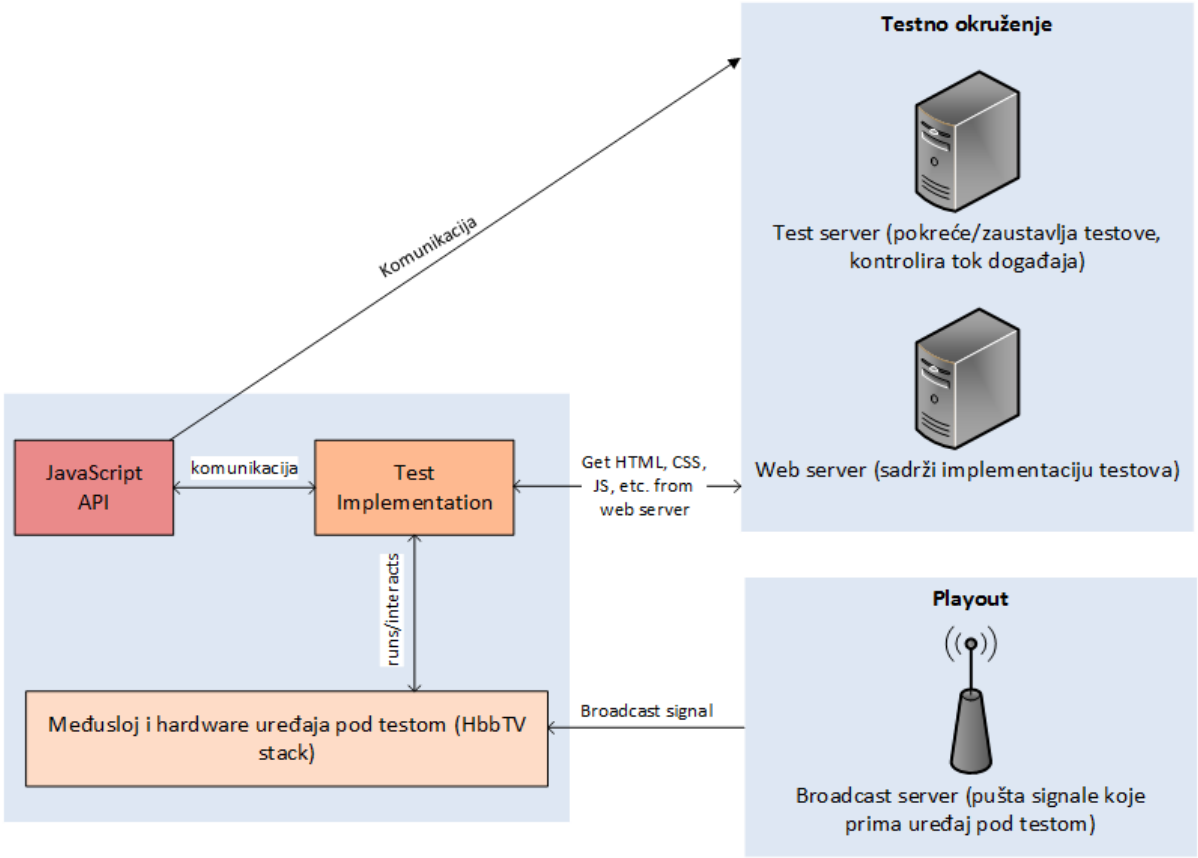
Ovdje ćemo opisati sljedeća sučelja :

- JavaScript API koji definira sučelje između Test Harness-a i DUT-a
- Skup XML datoteka koje definira kako Test Harness interpretira testni slučaj. To dozvoljava definiranje i kontrolu DVB *playout*-a potrebnog za inicijaciju testa.

Kreator testnog slučaja može napraviti testni slučaj koji se može izvesti na bilo kojem HbbTV uređaju kompatibilnim s Test Harness-om, ako su sučelja definirana. Također definicija sučelja omogućava implementaciju više instanci Test Haress-a. Test Haress-i mogu imati različite razine autonomnosti, ali svi moraju biti kompatibilni s testnim slučajevima pridodanim sučelju.
Na slici 2.7 je prikazana komunikacija uređaja i Harness-a.



Slika 2.7 Pregled JS API komunikacije između DUT i Harnessa



Slika 2.8 Detaljni pregled JS API komunikacije između DUT i Harnessa

Raspored API-a opisanih u ovom dokumentu je napravljen tako da dozvoljava visok postotak automatizacije, nema potrebe za HbbTV Test okruženje da bude posve automatizirano. HbbTV JavaScript API-ji su zbog toga dizajnirani tako da je moguće ručno upravljati svakim testom. Postoji mogućnost višestruke implementacije pojedinog testa što dozvoljava kombinaciju testnih slučaja kako bi se napravilo kompletno HbbTV (automatizirano) testno okruženje.

JavaScript API je podijeljen u 3 dijela vidljivo na slici 2.8:

1. API-ji koji komuniciraju s testnim okruženjem. Oni obavještavaju testno okruženje o trenutnom statusu testa.
2. API-ji koji komuniciraju s DUT-om. Ovaj dio može biti implementiran od strane DUT proizvođača ili od nekog drugog.
3. API-ji koji komuniciraju s *Playout* okruženjem. *Playout* okruženje je odgovorno za slanje ispravne AIT tablice DUT-u kako bi se test mogao pokrenuti na tom DUT-u.

2.6.1 API komunikacija s testnim okruženjem

Navesti će nekoliko funkcija koje se korite u JS API-u i dati primjer komunikacije.

Ime funkcije	Kratak opis funkcije
getPlayoutInformation()	funkcija vraća informacije o trenutnom *playout*-u
endTest()	funkcija ukazuje na to da je testni slučaj završen
reportMessage()	funkcija vraća poruke s informacijama ka Test Harness-u
waitForCommunicationCompleted()	funkcija provjerava da li su svi pozivi uspješno dostavljeni serveru
manualAction()	funkcija nalaže test operatoru da izvrši akciju

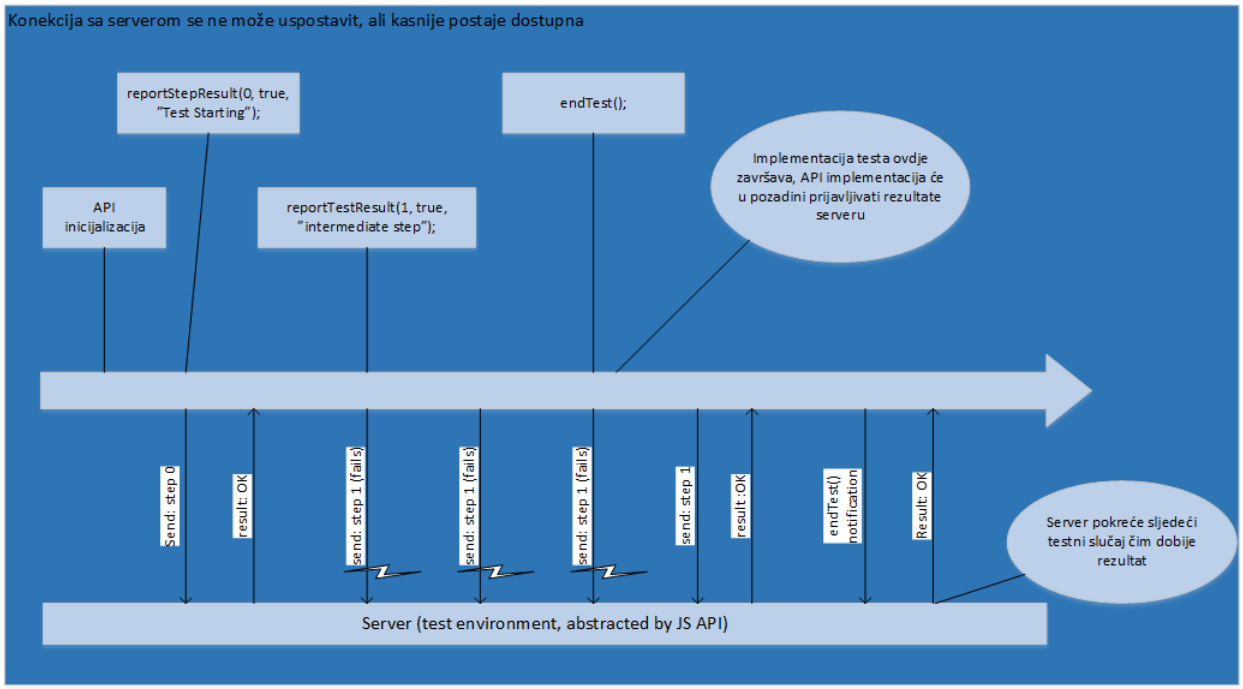
2.6.1.1 Prijava rezultata kada nema mrežne povezanosti

Pri pozivu funkcije mrežna komunikacija može biti nedostupna, tako da bi ova funkcija trebala biti implementirana asinkrono. U svakom slučaju implementacija mora osigurati da rezultati svih koraka budu prijavljeni serveru odmah kada se ponovo uspostavi mrežna komunikacija u onom redu u kojem su nastali. Mogu biti sljedeće implementacije:

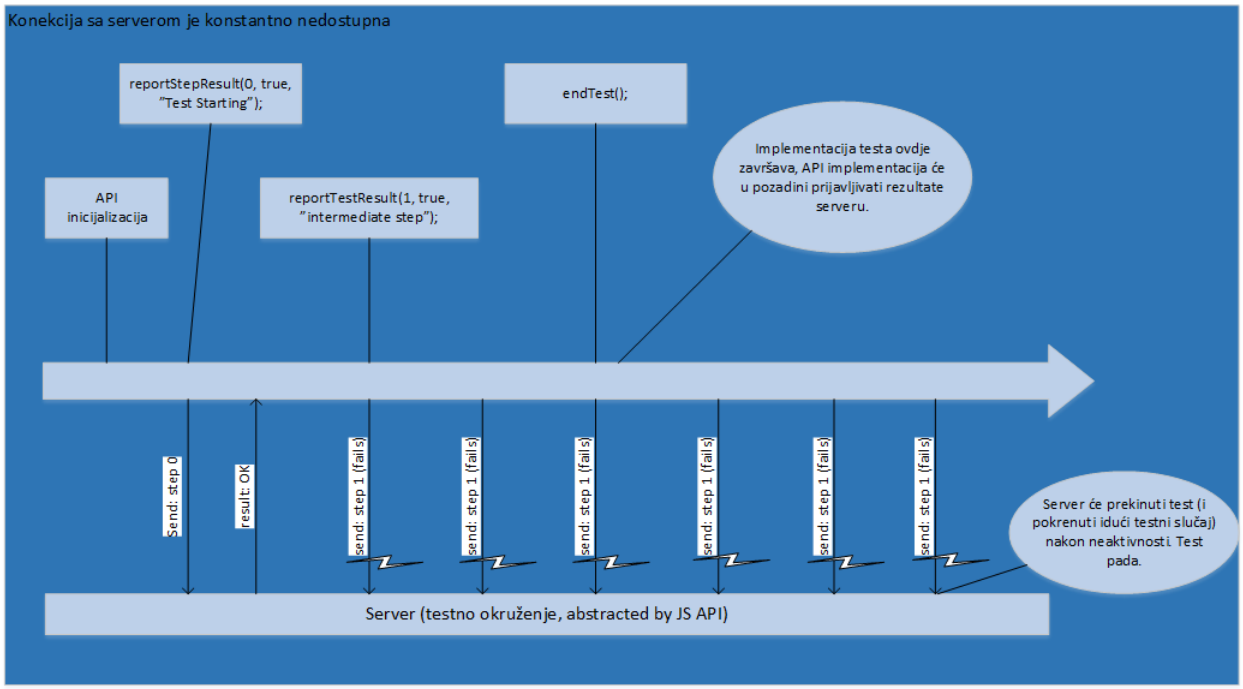
Primjer 1. Sinkrona komunikacija sa serverom putem ne- mrežnog puta (npr. serijski linijski priključak)

Primjer 2. Asinkrona komunikacija sa serverom gdje su svi pozivi spremljeni u FIFO (eng. first in, first out) red, i jedan po jedan se prijavljuju serveru. Ako komunikacija nije moguća, privremeno nedostupna prikazana na slici 2.9 ili konstantno nedostupna prikazana slikom 2.10, JS API implementacija će u pozadini konstantno pokušati prijaviti rezultate koraka u redu. To je razlog zašto izvršitelj testa mora osigurati da je mrežna komunikacija dostupna na kraju testa.

Sljedeći dijagrami pokazuju komunikaciju sa serverom u slučaju asinkrone komunikacije sa serverom.



Slika 2.9 Komunikacija između DTU i Test Harnessa kada je mreža privremeno nedostupna



Slika 2.10 Komunikacija između DTU i Test Harnessa kada je mreža trajno nedostupna

2.6.2 API interakcija s uređajem pod testom

Sljedeće funkcije se koriste za komunikaciju s uređajem pod testom.

Ime funkcije	Kratak opis funkcije
initiatePowerCycle()	funkcija pokreće "power cycle" (paljenje gašenje u danim intervalima) DUT-a
sendKeyCode()	funkcija zahtjeva "key code" (ID tipke) koji ce biti poslan DUT-u
analyzeScreenPixel()	funkcija analizira trenutni zaslon i provjerava da li traženi pixel screenshot-a odgovara danoj referentoj boji

analyzeScreenExtended()	funkcija analizira trenutni zaslon i obavlja detaljnu provjeru na trenutnom sadržaju zaslona
analyzeAudioFrequency()	funkcija analizira trenutni audio i obavlja frekvencijsku provjeru nad tim podacima
analyzeAudioExtended()	funkcija analizira trenutni audio i obavlja detaljnu provjeru nad tim podacima
analyzeVideoExtended()	funkcija analizira trenutni video i obavlja detaljnu provjeru nad tim podacima
analyzeManual()	funkcija naređuje test operatoru da uradi analizu opisanu u danim parametrima i snimi rezultate
selectServiceByRemoteControl()	funkcija zahtjeva odabir usluge u odnosu na dobijenoj sekvenci “key code-ova” koji su poslani DUT-u
sendPointerCode()	funkcija zahtjeva od pokaznog uređaja da se pomjeri na određenu lokaciju zaslona DUT-a
moveWheel()	funkcija zahtjeva od uređaja s točkićem da se pomjeri relativno u odnosu na trenutni položaj

2.6.3 API komunikacija s Playout okruženjem

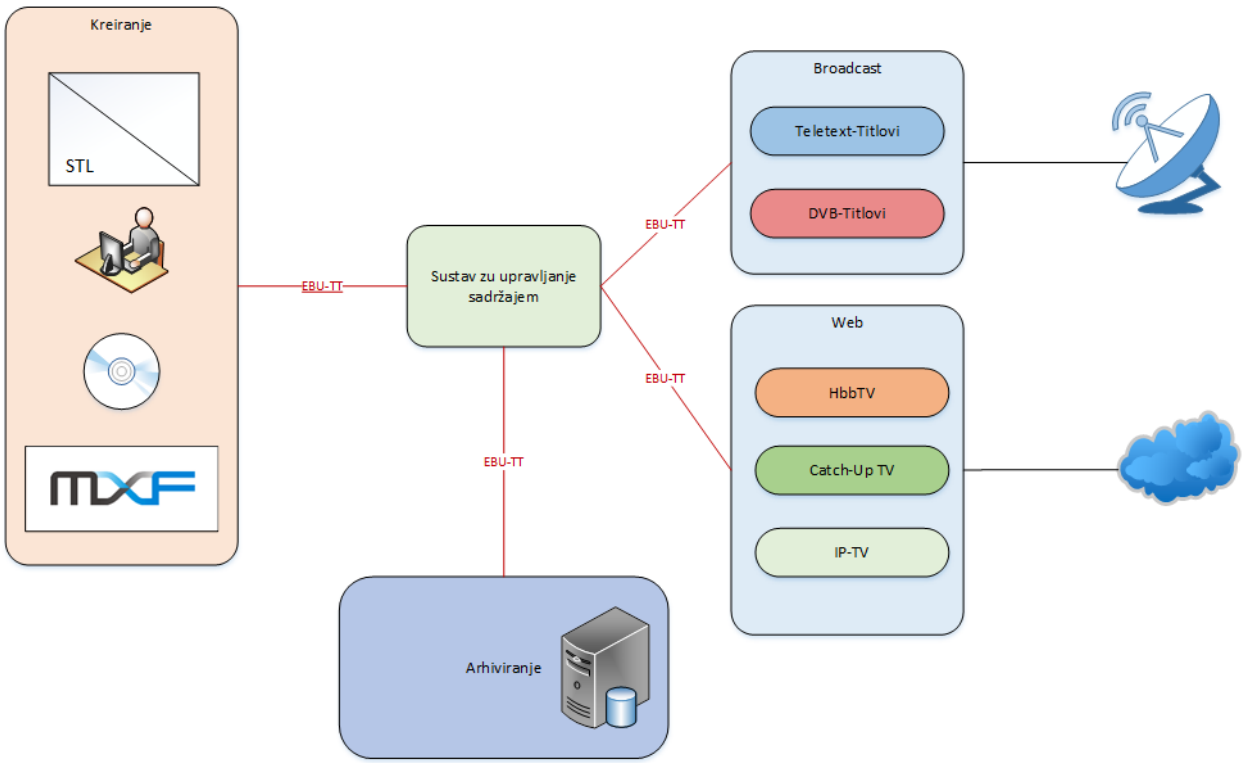
Sljedeće funkcije se koriste za komunikaciju s Playout okruženjem.

Ime funkcije	Kratak opis funkcije
changePlayoutSet()	funkcija mijenja trenutni *playout* (time može onemogućiti mrežnu konekciju)

setNetworkBandwidth()	funkcija ograničava maksimum podataka aplikacije dozvoljenih na mrežnom sučelju uređaja pod testom.

3. EBU-TT

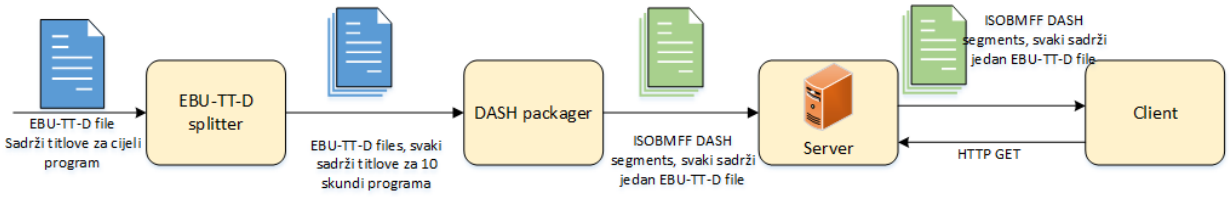
Razvojem televizije i većih rezolucija ekrana razvila se potreba za novim načinom prikaza titlova. Nastala su nova očekivanja za prikaz titlova na novi više korisnički prijateljski prikaz. Zahtjeva se novi format titlova zasnovan na XML-u koji omogućuje da se zadrže karakteristike za vrijeme kreacije i prijenosa. EBU-TT je format opće namjene za prijenos titlova i podržava unicode znakove. Osim za prijenos također se koristi za arhiviranje i produkciju. Njime su definirani parametri za XML dokument kojim se prenose titlovi. U XML su zapisani svi parametri i opcije titlova koji se prenose. Da bi se povećala kvaliteta rada kreatora titlova potreban je bolji autonomni proces, koji omogućava kreatoru titlova da poveća njihovu vrijednost koristeći svoja znanja o kontekstu, znanje kulture i specijalne vještine. Također su vrlo važni pouzdani mehanizmi razmjene za kreiranje titlova, koje se često odvija u vanjskoj produkcijskoj kući ili kod kuće koristeći veliki spektar različitih platformi i aplikacija. Na slici 3.1 je prikazan radni tijek titlova.



Slika 3.1. Radni tijek titlova u EBU-TT okruženju [2]

EBU-TT je namijenjen kao format opće namijene za titlove i podržava unicode znakove. Dobra praksa je provjeriti sukladnost dokumenata s EBU-TT specifikacijom, osobito na mjestima razmijene, ali će vjerojatno dodatne provjere biti zahtijevane na određenim mjestima u radnom toku. EBU-TT koristi podskup vokabulara sadržanog u WC3 *Timed Text Markup Language* 1 (TTML1) (*Second Edition*) , tako zvanog TTML 1.0. Ovaj WC3 standard je prethodno bio znan kao Timed *Text* (TT) *Authoring Format* 1.0 – *Distribution Format Exchange Profile* (DFXP). EBU-TT je namijenjen da ograniči značajke ponuđene od strane TTML 1.0, osobito s namjerom da EBU-TT bude više pogodan za upotrebu u video sadržajima emitiranih prijenosnom tokom i web video aplikacijama. Ispravan EBU-TT dokument je, po definiciji TTML 1.0, ispravan WC3 TTML dokument. Međutim moguće je napraviti ispravan WC3 TTML dokument koji nije ispravan EBU-TT dokument. Kao i EBU-TT specifikacija, SMPTE standard ST 2052-1:2010 (SMPTE-TT) je također izveden iz TTML-a. Ispravan EBU-TT dokument je, po definiciji SMPTE-standarda, ispravan SMPTE-TT dokument. Međutim moguće je napraviti ispravan SMPTE-TT dokument koji nije ispravan EBU-TT dokument. Potrebno je napomenuti da se rad neće baviti drugim standardima izvan EBU-TT.

EBU-TT format definira ograničenja instanci XML dokumenata. Ispravan EBU-TT XML dokument mora se pridržavati općih ograničenja i strukture dokumenta definiranih u EBU-TT specifikaciji.



Slika 3.2. Potpuni EBU-TT-D lanac

Da bi se pustio testni programski tok, potrebno je napraviti MPEG (eng. Moving Picture Experts Group) DASH (eng. Dynamic Adaptive Streaming over HTTP) klijent koji razumije EBU-TT-D datoteke titlova i može postaviti titlove preko puštenog videa. Kada je klijent u mogućnosti renderirati EBU-TT-D datoteke titlova direktno, potrebno je dodati mogućnost da izvlači datoteke

titlova iz ISOBMFF pakiranja u kojima su oni poslani u MPEG DASH-u. Potpuni lanac titlova je prikazan na slici 3.2.

3.1 XML titlovi

Titlovi su spremljeni u XML datoteku koja sadrži sve informacije o titlovima, interakciju među njima, te same titlove. Sadrži sve regije titlova koje se parsiraju po tagovima da bi se dobile tražene opcije titlova. U nastavku će biti da pregled na najvažnije.

3.1.1 tt:tt

Glavni element titlova, svaki EBU-TT dokument mora počinjati ovim elementom. Ovdje su definirane sve inicijalne postavke prikaza titlova te sadrži sve ostale elemente. Neki od segmenata su: *ttp:timeBase* - odabir vremenske baze, *ttp:frameRate, ttp:frameRateMultiplier, tp:clockMode, ttp:cellResolution* – veličina ćelije, *xml:lang* – jezik.

3.1.2 tt:head

Kontejner u koji su spakirani svi elementi koji se odnose na izgled titlova, stilovi i regije. Definirani su specifični stilovi i regije koje će poslije elementi iz „body“ taga kasnije pozivati.

3.1.3 tt:styling

Kontejner za sve informacije o stilu titlova. Sadrži atribute bitne za stil kao *fontStyle,* fontWeight, te različite stilove definirane u *tt:style*.
tt:style sadrži detaljne informacije o stilovima. Može postojati više adaptacija koje se razlikuju po ID - u te nazivu. Sadrži elemente kao: *style* koji mu omogućuju da naslijedi opcije drugih već kreiranih stilova, *tts:direction* smjer ispisa titlova, *tts:fontFamily* vrsta fonta, *tts:color* boja titlova, *ts:backgroundColor* boja pozadine i druge.

3.1.4 tt:layout

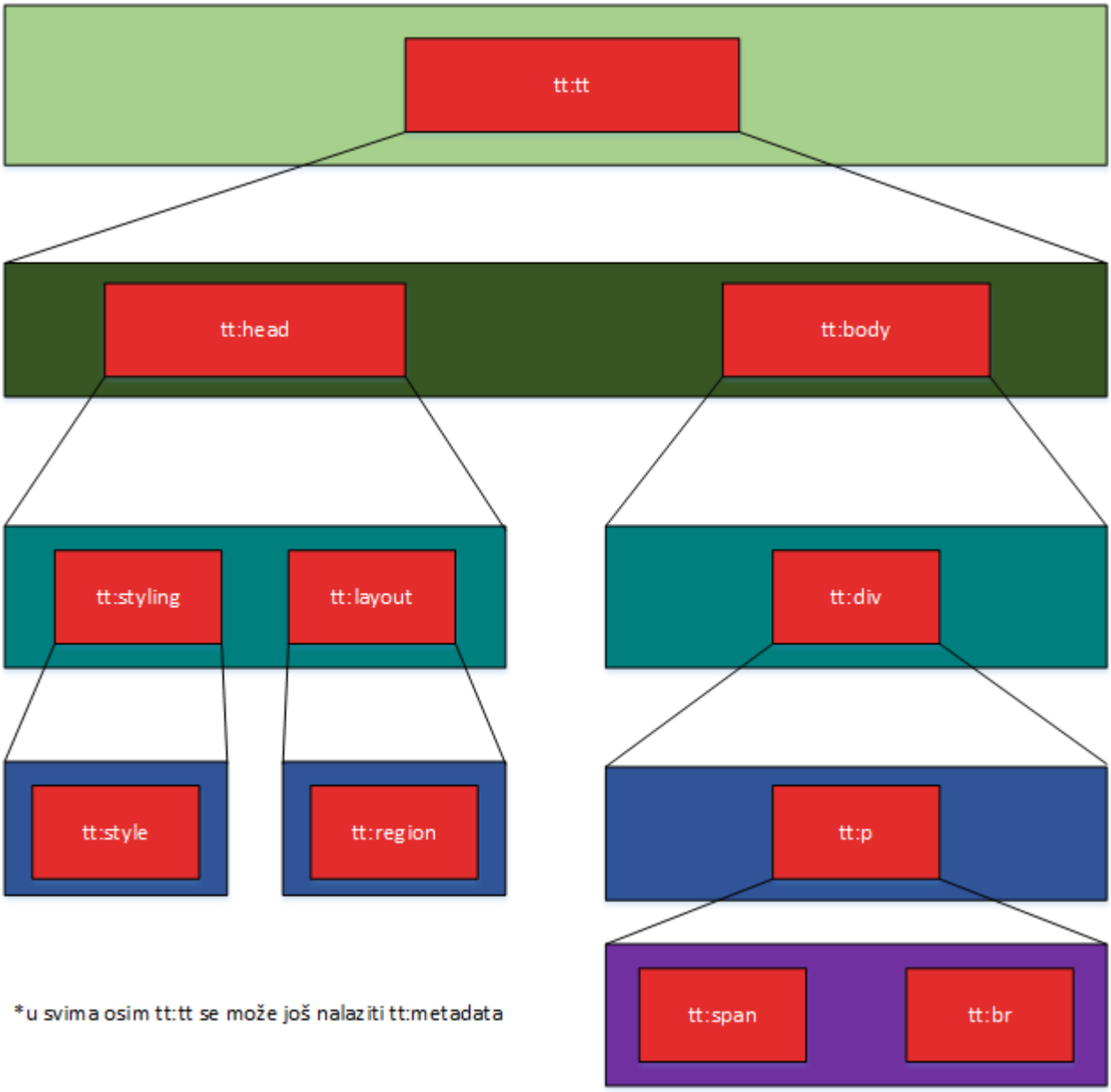
Kontejner za sve elemente regija titlova. Može sadržavati nekoliko *tt:region* elemenata u kojima su definirane regije.
tt:region sadrži detaljne informacije o regijama koje se razlikuju po ID-u i nazivu. Sadrži atribute: *tts:origin* definira početni gornji lijevi kut regije, *tts:extent* Definira konačni donji desni kut regije, *style* definira prethodno kreirani stil koji će se primijeniti na regiju, *tts:showBackground* određuje hoće li se pozadina prikazivati kada nema prisutnih titlova i neke dodatne opcije.

3.1.5 tt:body

Kontejner za titlove i informacije o vremenima prikaza. Za prikaz u regijama i stilove prikaza koristi reference na prethodno kreirane *tt:style* i *tt:region*. Sadrži elemente u kojima se nalaze titlovi *tt:div, tt:p, tt:span.*

3.1.6 tt:div

Kontejner za tekstualni sadržaj. U njemu se nalaze elementi koji sadrže tekst titlova *tt:p, tt:span* i *tt:br*. Može postojati više elemenata koji se razlikuju po *xml:id*. Dodatni atributi koje može sadržavati su *region* odabir prethodno kreirane regije, *style* prethodno kreirani stil, *xml:lang* jezik. Svi atributi se odnose i na niže elemente unutar taga. Također može sadržavati dodatne *tt:div* elemente u sebi.
tt:p podelement *tt:div* u kojem su zapisani tekstualni podaci titlova. Može postojati više elemenata koji se razlikuju po *xml:id*. Ovdje su spremljeni tekstualni podaci titlova te druge karakteristika kao: *begin* početak prikaza, *end* kraj prikaza, *region* regija prikaza, *style* stil prikaza. Također može sadržavati više instanci *tt:span* i *tt:br*.
tt:br označava početak novog reda.
tt:span također sadrži tekstualne podatke titlova te može sadržavat opcije kao *begin* početak prikaza, *end* kraj prikaza, *region* regija prikaza, *style* stil prikaza. Potrebno je napomenuti da ako više elemenata ima isti atribut prednost ima onaj na nižoj razini. Dodatni razlog korištenja je primjena posebnog stila na samo jedan dio titla, npr. jednu riječ.

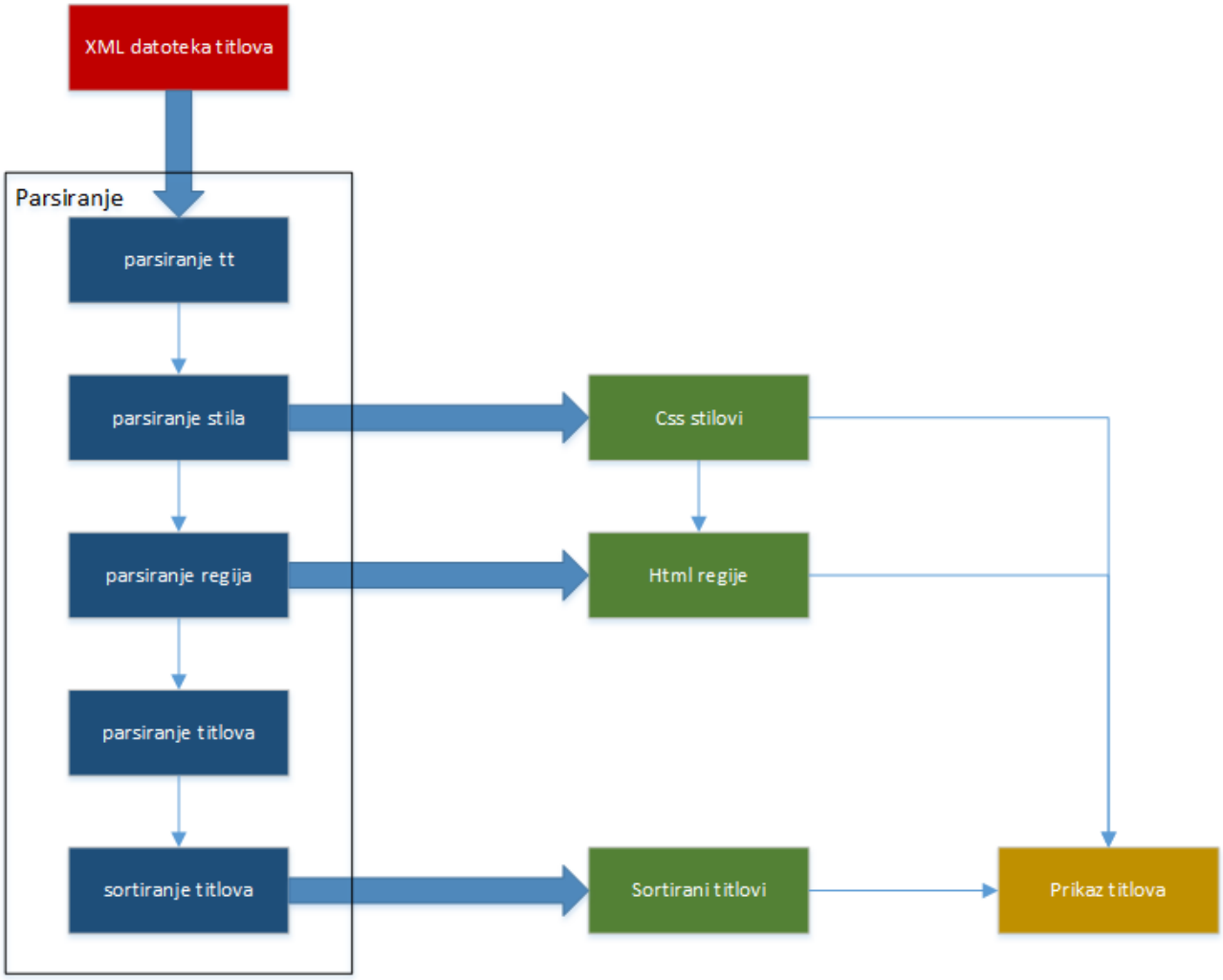


Slika 3.3. Hijerarhijski model glavnih elemenata EBU-TT dokumenta

Slika 3.3 prikazuje glavne elemente dokumenta. Za detaljan opis svih mogućih elemenata i njihove detaljne opise pogledati EBU-TT_SPEC [2].

4. EBU_TT Intepreter

Skripta za tumačenje EBU-TT-D titlova napisana u JavaScript-u. Tok obrade je prikazan na slici 4.1.

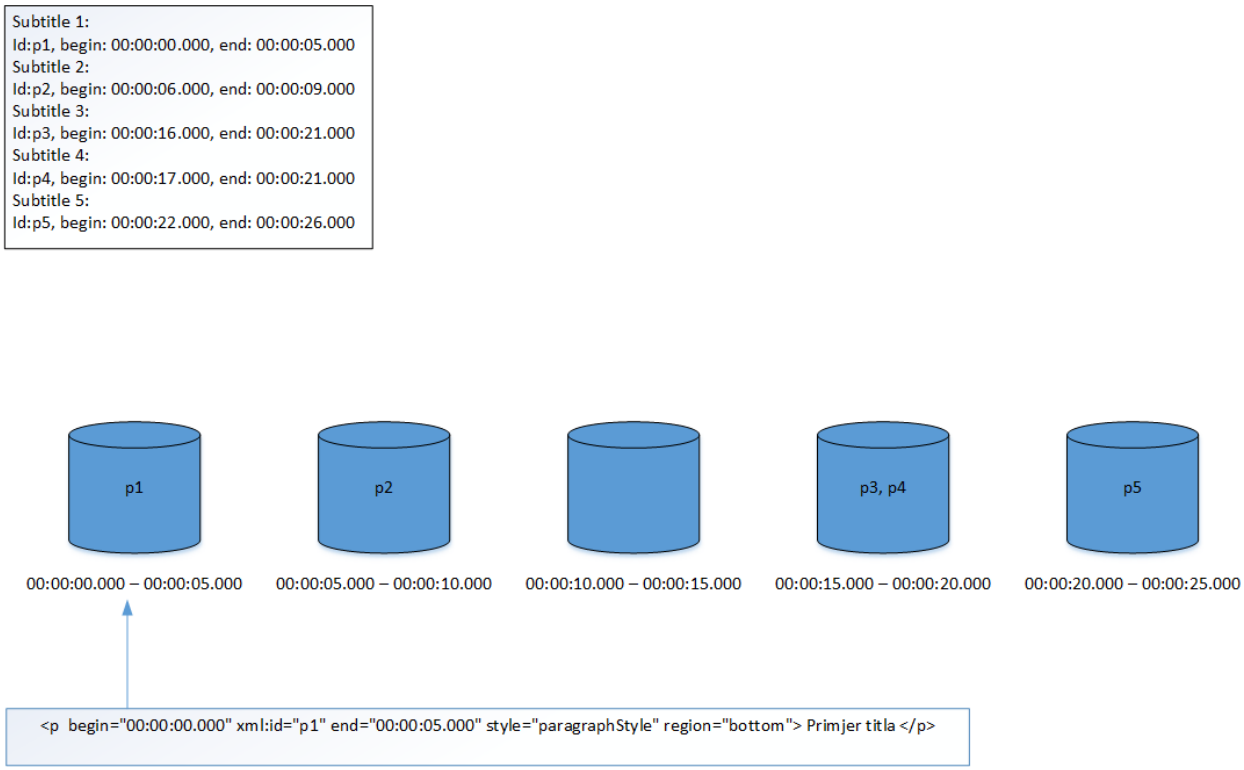


Slika 4.1 Tok obrade titlova

Rješenje je implementirano kao JavaScript biblioteka, što nam omogućava da ga ugradimo u bilo kakvu medijsku platformu koja se izvršava u web okruženju. Integrira se sa danim HTML5 video player-om, prateći njegov *timeline*, dohvaća trenutno vrijeme videa, kako bi se titlovi prikazali preko video sadržaja u odgovarajućim trenucima.

Bucket struktura je upotrjebljena da bi se efektivno mogla vršiti pretraga titlova koje treba prikazati u odgovarajućim trenucima. U intervalima od 5 sekundi, smještaju se titlovi koji se tada trebaju

prikazati, te se pri provjeri da li se neki titl treba prikazati, pretražuje samo u okviru odgovarajućeg *bucket*-a umjesto cijele liste svih titlova. Primjer bucket strukture je vidljiv na slici 4.3.



Slika 4.2. primjer rada BucketSort-a

XML datoteka sa EBU-TT titlovima se dobavlja putem AJAX-a i prosljeđuje interpreteru, zajedno sa referencom na HTML5 video *player*, nakon čega slijedi inicijalizacija interpretera i parsiranje datoteke. Prva funkcija od koje se počinje je EBU_Builder i ona poziva sve ostale funkcije. Prva funkcija koju poziva je ttParser koja sprema atribute iz <tt> sekciju u prethodno kreirani objekt nazvan tt koji sadrži sve moguće varijable koje se mogu nalaziti u <tt> sekciji. Funkcija ttParser koristi getElementByTagNameNS kojom dohvaća "tt" atribute te prolazi kroz njih u for petlji te sprema vrijednosti atributa u odgovarajuće elemente ttItem objekta i pomoću switch-case kreira odgovarajuće radne parametre. Povratna vrijednost je ttItem objekt koji se sprema u tt objekt EBU_Builder.

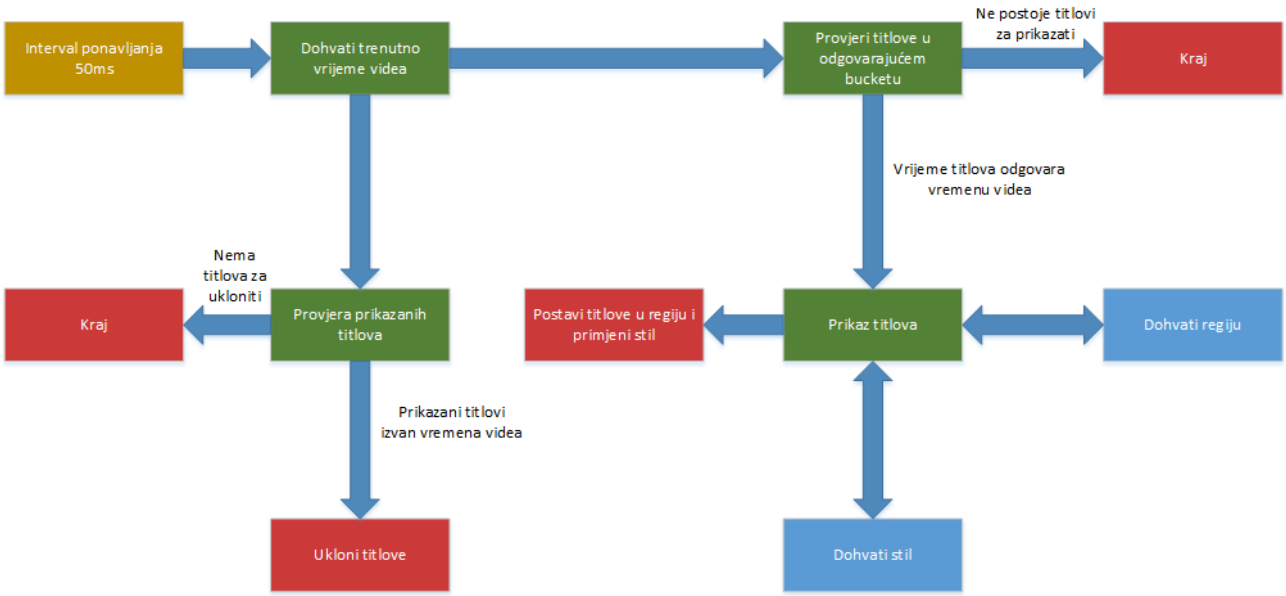
Sljedeće je potrebno učitati stilove. Za to se prvo poziva funkcija fontParser koje parsira fontove. Funkcijom getElementByTagNameNS dohvaća "font" atribute i prolaskom for petlje sprema atribute u objekt. Pomoću *switch-case* provjerava svaki atribut i kreira css tekst koji sprema u

varijablu absStyle. Nakon što je prošao kroz sve atribute kreira css stil pozivom funkcije createFontCss koja vrijednost varijable absStyle pretvara u HTML CSS stil i postavlja na stranicu. Ako je postojala definicija fonta ona je sada kreirana pa nakon toga slijedi kreiranje stilova koji će se primjenjivati za regije i titlove. Pokreće se styleParser te kao i prethodnim funkcijama prolazi se kroz sve atribute za svaki stil te pomoću *switch-case*-a se kreira odgovarajuća CSS klasa za svaki stil definiran u XML dokumentu.

Kada postoje kreirani stilovi potrebno je napraviti regije u kojima će se prikazivati titlovi. Pronalazi se dio XML označen s "region" i parsira se. For petlja prolazi kroz sve atribute i kreira se HTML regija za svaku regiju definiraju XML-om u koje će se naknadno stavljati titlovi.

Nakon što imamo stilove i regije može se početi s parsiranjem sadržaja titlova.

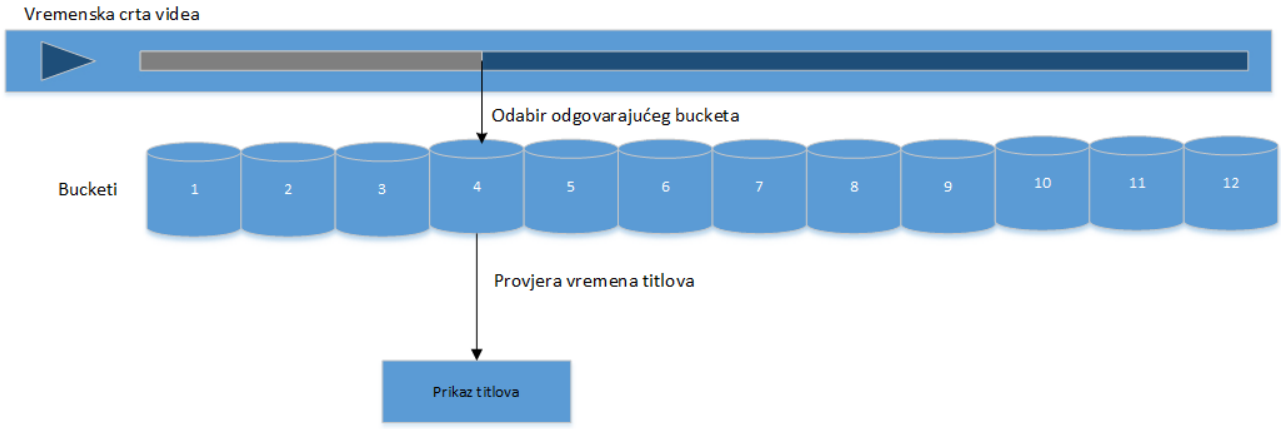
Pronalaze se svi „p" tagovi i parsiraju se. Iz njih se izvlače svi titlovi i sortiraju se u buckete. Tako spremljeni titlovi su spremni za prikaz na ekranu i mogu se koristiti u daljnjim funkcijama.



Slika 4.3 Prikaz titlova

Završne funkcije su za prikaz i uklanjanje titlova s ekrana. „SubtitleDisplay" se koristi za prikaz titlova, provjerava se trenutno vrijeme i uspoređuje s vremenima titlova u bucketima, ako je vrijeme odgovarajuće titlovi se prikazuju i spremaju kao trenutno prikazani titlovi. U isto vrijeme se odvija funkcija „removeSubtitles" koja provjerava vremena trenutno prikazanih titlova s trenutnim vremenom videa i ako su vremena različita uklanja titlove. Ovo se odvija konstantno dok traje

video. Prikaz titlova se odvija prema slici 4.3, na slici 4.4 je vidljiv odabir bucketa te se primjer izgleda titla može vidjeti na slici 4.5.



Slika 4.4 Odabir odgovarajućeg bucketa



Slika 4.5 Primjer izgleda titla

4.1 Popis funkcija

Navedene su funkcije korištene u EBU-TT intepreteru te opis njihovog rada.

Ime funkcije	Kratak opis funkcije
EBU_Builder	• Počinje se izvršavanje skripte • Otvara i iščitava xml datoteku • Poziva ostale funkcije
ttParser	• Kreira varijable za konfiguracijske parametre • Pronalazi sve elemente s tagom „tt“ • Vrti for petlju kroz sve elemente te sprema vrijednosti elemenata u odgovarajuću varijablu • Sadrži funkcije: createTtParameters, o Poziva se za svaki element o *Case* petlja koja kreira odgovarajuće radne parametre • parseDimension pretvara tekstualni oblik dimenzija u odgovarajući oblik upotrebljiv skripti
fontParser	• Kreira varijable za fontove • Pronalazi sve elemente s tagom „font“ • Vrti for petlju kroz sve elemente te sprema vrijednosti elemenata u odgovarajuću varijablu • Sadrži funkcije: createFont, o Poziva se za svaki element o *Case* petlja koja kreira odgovarajuće CSS varijable • createFontCss o Dodaje parsirane font varijable kao CSS style u stranicu • appendStyleText o poziva se za postavljanje varijabli u oblik razumljiv CSS-u
styleParser	• Kreira varijable za stilove • Pronalazi sve elemente s tagom „style“

	• Vrti for petlju kroz sve elemente te sprema vrijednosti elemenata u odgovarajuću varijablu
	• Sadrži funkcije: createStyle,
	  o Poziva se za svaki element
	  o *Case* petlja koja kreira odgovarajuće CSS varijable
	• createStyleCss
	  o Dodaje parsirane style varijable kao CSS style u stranicu
	• appendStyleText
	  o poziva se za postavljanje varijabli u oblik razumljiv CSS-u
	• getColor
	  o Poziva se kada je potrebni iščitati boju iz dokumenta
	  o Pretvara oblik iz XML dokumenta u oblik razumljiv CSS-u

createInitaalCtyleValues	• Kreira inicijalni stil za elemente koji nemaju dodijeljen posebni stil

regionParser	• Kreira varijable za regije
	• Pronalazi sve elemente s tagom „region“
	• Vrti for petlju kroz sve elemente te sprema vrijednosti elemenata u odgovarajuću varijablu
	• Sadrži funkcije: createRegionStyle,
	  o Poziva se za svaki element
	  o *Case* petlja koja kreira odgovarajuće CSS varijable
	• createRegion
	  o Kreira potrebne regije na kojima će se kasnije prikazivati titlovi
	• createRegionCss
	  o Dodaje parsirane style varijable za regije kao CSS style u stranicu
	• parseDimension
	  o pretvara tekstualni oblik dimenzija u odgovarajući oblik

	upotrebljiv skripti • appendStyleText o poziva se za postavljanje varijabli u oblik razumljiv CSS-u
pParser	• Kreira varijable za titlove • Pronalazi sve elemente s tagom „p“ • Vrti for petlju kroz sve elemente te sprema vrijednosti elemenata u odgovarajuću varijablu • Sprema vremena početka i kraja titla i njihov sadržaj • Sadrži funkcije: spanParser, o Funkcija se poziva za span childNode od p o Parsira span titlove i sprema vremena i sadržaj titlova o Poziva sama sebe ako postoje dodatni childNode span-a • bucketSort o sortira parsirane titlove u buckete o trajanje jednog bucketa je 5s o ovisno o vremenu početka titlovi se spremaju u bucket određenog indeksa
subtitleDisplay	• Funkcija za prikaz titlova na ekranu • Poziva se u periodičkim intervalima od 50ms • Dohvaća trenutno vrijeme videa ili lokalno vrijeme ovisno o postavkama titlova • Ovisno o vremenu dohvaća odgovarajući bucket • Uspoređuje trenutno vrijeme s vremenima titlova u bucketu i ako odgovaraju dodaje titlove u odgovarajuću regiju i prikazuje ih, te sprema informaciju o trenutno prikazanim titlovima • Provjerava trenutno prikazane titlove i ako vremena ne odgovaraju trenutnom vremenu poziva removeSubtitles • Sadrži funkciju: spanDisplay o Služi za prikaz span childNode-ova o Poziva sama sebe ako ima više razina childNode

removeSubtitles	• Uklanja traženi titl s ekrana • Ano nema prikazanih titlova postavlja regiju u skrivenu ako je tako postavljeno
removeNamespace	• Funkcija koja se koristi u svim parsiranjima za uklanjanje namespace odnosno prefiksa ispred imena varijabli, ako postoji

5. Testovi

Da bi se provjerila ispravnost nekog uređaja potrebno je napraviti testne slučajeve.

Testovi se sastoje od:

1. XML datoteke s opisom testa
    a. Naslov testa (title)
    b. Opis testa (assertionText)
    c. Opis testnih koraka (testStep)
    d. Zahtjev za prolaz testa (passCriteria)
    e. Povijest (history)
    f. Tko je pridonio (contributor)
2. HTML5 datoteka koja je zapravo test koji se izvršava u pretraživaču
3. T2 Unit js koji se koristi za testiranje testa
4. T2 Unit json testni plan po kojem se izvršava T2 testiranje

5.1 Opis izrade testa

Svi testovi imaju istu proceduru izrade tj. prolaze kroz iste korake izrade.

Pri izradi testnog slučaja provode se sljedeći koraci:

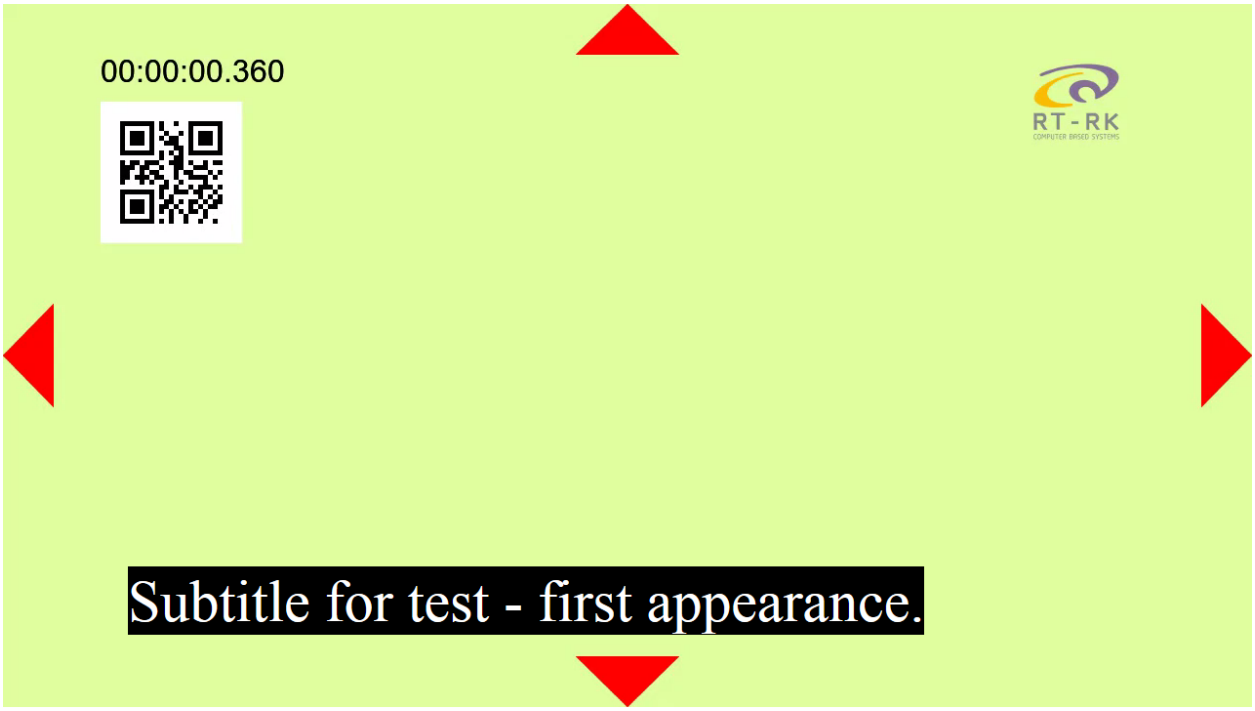
1. Dobivanje XML datoteke testa
2. Kreiranje Html5 datoteke testa prema zadatku upisanom u assertionText
3. Provjera ispravnosti testa
4. Kreirati js datoteku za svaki mogući rezultat testa potreban za T2
5. Napraviti testni plan za izvršavanje testa
6. Testiranje

5.2 Urađeni testovi

Urađeni su za testovi za titlove, pokazivač, tipke na daljinskom i html5 testovi.

Testovi za titlove (SUB)

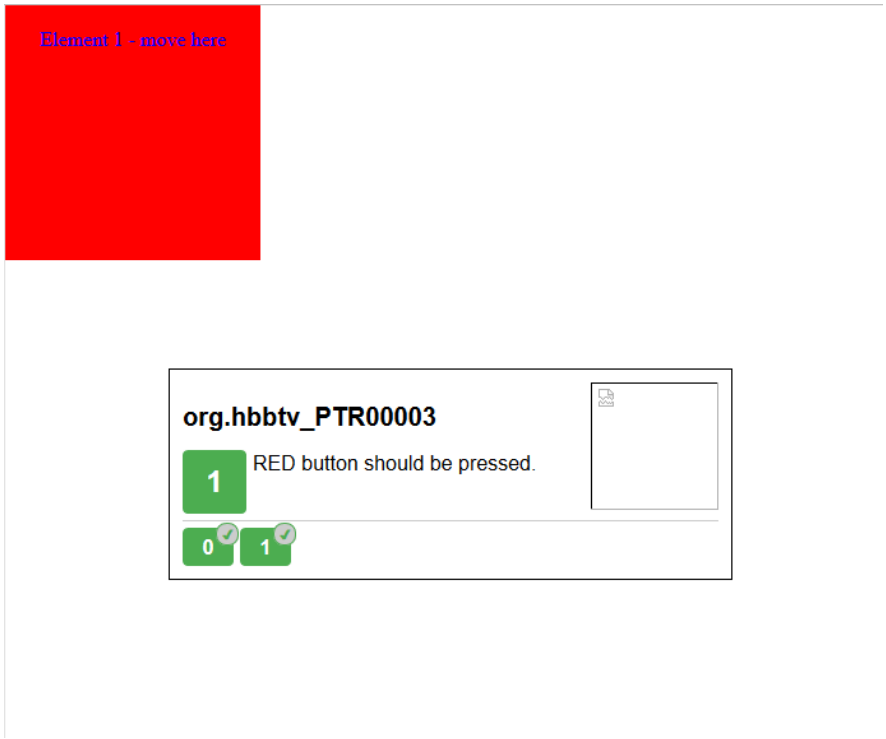
Provjerava se ispravnost prikazanih titlova na uređaju. Neki primjeri što se može testirati je sinkronizacija titlova i videa, ispravni prikaz ne standardnih znakova, ispravan prikaz regije, stila, boje ili fonta. Iz originalne xml datoteke testa iščitava sa zadatak testa pod assertionText. Prema opisu testa kreira se xml datoteka titlova, koji se trebaju prikazati u testu. Pomoću prethodno kreiranog EBU-TT intepretera za titlove prikazuju se titlovi na testnom videu i uzima se snimak ekrana kako treba to izgledati. Zatim se kreira html5 datoteka sa testnim koracima i prethodno spremljenom referentnom slikom koji će se prikazivati testeru. Pri izvršavanju testova na uređaju pod testom uspoređuje se pušteni testni video s titlovima s referentnom slikom. Slika 5.1 prikazuje referentnu sliku titla. Ako su slike jednake uređaj je prošao test inače je pao.



Slika 5.1. Referentni prikaz titlova

Testovi za pokazivač (PTR)

Provjerava se generiranje ispravnih događaja vezanih uz pokazivač. Neki primjeri koji se mogu testirati su događaji: mouseenter – ulaz pokazivača u regiju, click – klik miša na regiju, dbclick – dvostruki klik miša, mouseleave – izlaz pokazivača iz regije, korištenje kotačića na mišu. Iz originalne xml datoteke testa iščitava sa zadatak testa pod assertionText. Kreira se html5 datoteka testa. Sadržava referentnu regiju na koju će se pomaknuti ili maknuti iz nje pokazivač. Traži se od testera da pomakne pokazivač u početni položaj i kreiraju se slušatelji događaja (*EventListener*). Traži se od testera da pomakne pokazivač na traženu lokaciju i provjerava se da li se generirao događaj i da li je ispravan događaj generiran. Ako je generiran ispravan događaj uređaj je prošao test, a ako nije pao je test. Na slici 5.2 je prikazan izgled testa.

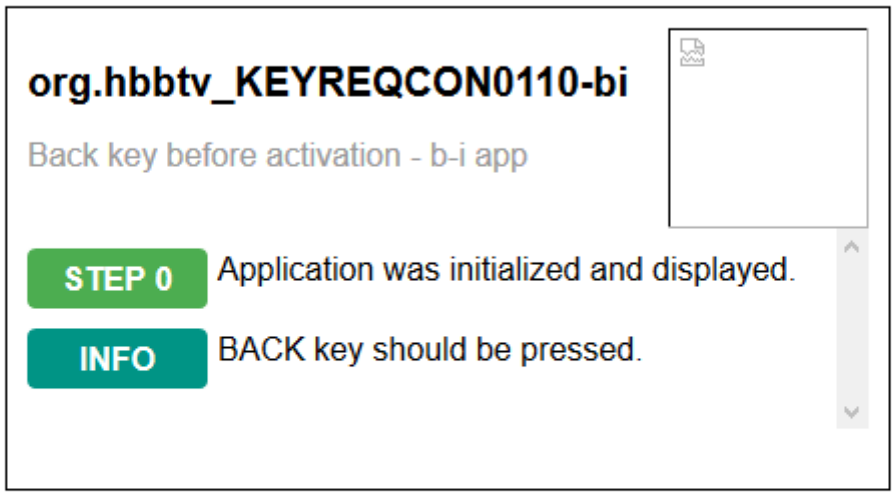


Slika 5.2. Referentni izgled testa za pokazivač

Testovi za tipke daljinskog (KEYREQCON)

Provjerava se generiranje ispravnih događaja vezanih uz daljinski upravljač. Neki primjeri koji se mogu testirati su generirani događaji za tipke *play*, *stop*, *pause*, *record*. Iz originalne xml datoteke testa iščitava sa zadatak testa pod assertionText. Kreira se html5 datoteka testa. Kreira se aplikacija koja osluškuje tipke sa daljinskog upravljača i kreiraju se slušatelji događaja (*EventListener*). Traži se od testera da pritisne potrebnu tipku i provjerava se da li je ispravna tipka pritisnuta, ako se

generira ispravan događaj test je prošao u drugom slučaju uređaj je pao test. Na slici 5.3 je prikazan izgled testa.



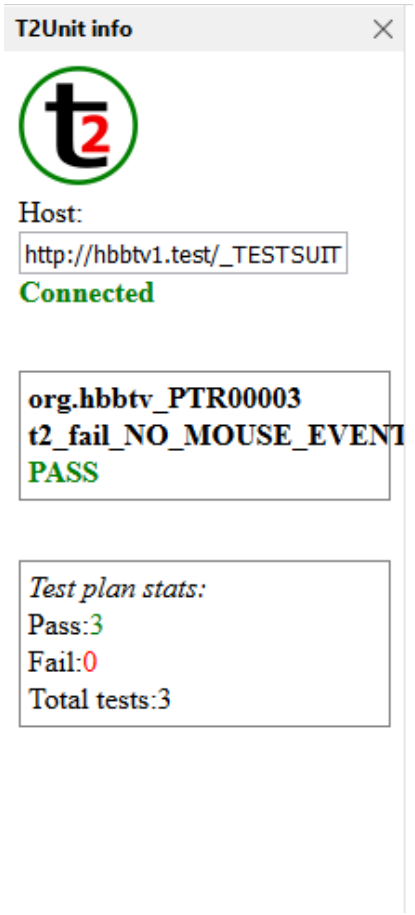
Slika 5.3. Referentni izgled testa za tipke daljinskog upravljača

Testovi za html5 (HTML5)

Provjerava se mogućnosti uređaja vezanih uz html5. Neki primjeri koji se mogu testirati su provjera roditeljske kontrole, ispravnost rada *download*-a, provjera ispravnih tagova u MPD DASH-u. Iz originalne xml datoteke testa iščitava sa zadatak testa pod assertionText. Kreira se html5 datoteka testa. Mora postojati prikladna mpd ili ts datoteka za test iz koje se iščitava traženi tag. Provjerava se ispravnost taga i ako je ispravan o očitan i ima pravu vrijednost uređaj je prošao test

5.3 T2 Unit

T2 uni služi za testiranje ispravnosti testova, tj. testovi za testiranje testova , odakle je i dobio naziv T2. Potrebno je napraviti nekoliko js datoteka , pri čemu svaka pokriva jedan mogući rezultat testa. Potrebno je napravit dovoljno js datoteka za svaki mogući rezultat testa. Js datoteke generiraju virtualne događaje i provjeravaju da li je rezultat testa jednak očekivanom. Ako je rezultat testa jednak očekivanom rezultatu za sve slučajeve test je prošao i ispravan, a u slučaju da postoji razlika u samo jednom koraku test nije ispravan. Kao primjer se može navesti testiranje testova za pokazivače. Prva js datoteka generira ispravan događaj (event) i očekuje da će test proći, dok druge dvije skripte očekuju da će test pasti tako što jedna ne generira događaj, dok druga generira krivi događaj. Potrebno je još imati json testni plan prema kojem se izvršavaju testiranja. U njemu se sadrže sve js skripte koje se trebaju izvršiti i redoslijed izvršavanja. Slika 5.4 prikazuje T2 dodatak koji daje informacije o uspješnosti testova.



Slika 5.4. T2 dodatak pretraživaču

6. Zaključak

Kako je razvoj tehnologije sve brži stari sustavi više nisu zadovoljavajući. Porasla je potreba ra integracijom televizije i internetskih usluga, što uzrokuje nastanak nove hibridne televizije, HbbTV. HbbTV je sve više rasteći standard sa sve većim zahtjevima i kvalitetom usluge. Naravno kako napreduje, televizija mora razviti i novi prikaz titlova koji će pratiti razvoj hibridnih tehnologija, EBU-TT. EBU-TT-D su napredniji i bolji prikaz titlova nego do sada korištenih normi. Zapis u xml-u omogućuje više mogućnosti titlova i bolju realizaciju. Intepreter za EBU-TT titlove je napisanu u javascriptu što omogućuje pokretanje na bilo kojem uređaju koji ima ugrađen internet pretraživač tako da je idealan za HbbTV uređaje, pošto svaki takav uređaj ima internet pretraživač. Intepreter ispravno prikazuje titlove i sve njihove opcije. Titlovi se prikazuju u ispravnim regijama i vremenima. Imaju ispravan prikaz boje, veličine slova i ostalih vizualnih stilova. Postoji mogućnost dodavanja vlastitih fontova. Iščitavaju se svi titlovi i sa nižih razina tj. childNode neovisno koliko ih ima. Titlovi su ispravno poredani u vremenu pomoću bucket sorta neovisno o njihovom originalnom položaju. Prikaz nije potpuno u stvarnom vremenu, nego se osvježava svakih 50ms što je dovoljno za ispravan prikaz.

Svi uređaju HbbTV-a moraju biti testirani da pružaju i pokrivaju sve mogućnosti propisane HbbTV specifikacijom, stoga je potrebno napraviti testne slučajeve koji će to provjeravati. Testovi za titlove traže interakciju i usporedbu slika od testera. Prethodnim intepreterom su snimljene referentne slike i onda ih tester uspoređuje, što znači da je rad na testeru uređaja. Testovi za pokazivač traže od testera samo da izvršava upute i zatim test sam provjerava rezultat i odlučuje o prolasku ili padu testa. Testovi za tipke daljinskog upravljača su također na isti princip gdje tester izvršava upute a test provjerava rezultat. Testovi za html5 se samo pokrenu i test sam provjerava ispravnost. Da bi neki uređaj bio proglašen HbbTV uređajem mora ispravno proći sve prethodne testove, čime je zadovoljio određene kriterije HbbTV standarda.

Literatura

[1] HBBTV_2_0_SPEC ETSI TS 102 796 V1.3.1 (2015-10)

[2] EBU-TT_SPEC TECH 3350 VERSION 1.1 Geneva September 2015

[3] http://www.bbc.co.uk/rd/blog/2015-01-subtitles-in-an-ip-world-1 [pristup ostvaren 1.9.2016]

Popis Kratica:

HbbTV - Hybrid Broadcast Broadband TV

Js – JavaScript

HTTP – Hypertext Transfer Protocol

API - Application programming interface (Aplikacijsko programsko sučelje)

XML – EXtensible Markup Language

CSS – Cascading Style Sheets

HTML5 - HyperText Markup Language 5

JSON – JavaScript Object Notation

MPD – Media Presentation Description

TS - transport stream

DASH - Dynamic Adaptive Streaming over HTTP

EBU - European Broadcasting Union

EBU-TT - EBU Timed Text

MHEG - Multimedia and Hypermedia Experts Group

IP - Internet Protocol

VOD - video on demand

EPG - elektronski programski vodič

DVB - Digital Video Broadcasting

A/V - audio/video

DSM-CC - Digital storage media command and control

FDP - File Download Protocol

TCP - Transmission Control Protocol

UDP - User Datagram Protocol

DUT - Device Under Test

TH - Test Harness

FIFO - first in, first out

MPEG - Moving Picture Experts Group

ISO - International Organization for Standardization

ISOBMFF - ISO base media file format

Sažetak

Osnovni zadatak diplomskog rada je bio napraviti intepreter koji će tumačiti XML datoteke EBU-TT titlova te napraviti testne slučajeve za testiranje HbbTV uređaja. Intepreter i testovi su napisani u JavaScript programskom jeziku tako da su pogodni za svaku platformu na kojoj se vrti neki pretraživač. Izrada rada je počela od najvažnijeg dijela, pisanja js skripte za intepreter koji se koristio za ostale testove za titlove. Osim tih napravljeni su još neki dodatni testovi za pokazivač, tipke daljinskog i html5, koji su također pisani u JavaScript programskom jeziku. Testovi su uspješni izvršeni na pretraživaču računala te se mogu implementirati na druge uređaje da se provjeri njihov rad. Svi oni skupa omogućuju testiranje HbbTV uređaja.

Ključne riječi: HbbTV, Test Harness, XML , HTTP, EBU-TT, interpreter, JavaScript, testni slučaj, verifikacija, parsiranje

Summary

The main task of the thesis was to create interpreter to interpret the XML files of EBU-TT subtitles and make the test cases for testing HbbTV devices. Interpreter and tests are written in the JavaScript programming language, so that they are suitable for each platform which runs a browser. Preparation work has begun from the most important part, writing js script for interpreter that was used for other subtitles tests. In addition to these, there are made even some extra tests for the pointer, buttons on the remote and html5, which are also written in the JavaScript programming language. The tests were successfully completed on the computer browser and can be implemented on other devices to check their work. All together they allow testing HbbTV devices.

Keywords: HbbTV, Test Harness, XML, HTTP, EBU-TT, interpreter, JavaScript , test case, verification, parsing

Životopis

Mario Hrbak je rođen 24.studenog.1992. u Srijemskoj Mitrovici u Srbiji. Pohađao Osnovnu školu Dore Pejačević u Našicama, te svih osam razreda završio s odličnim uspjehom. Nakon završene osnovne škole 2007.g. upisuje, Srednju školu Isidora Kršnjavoga također u Našicama, smjer tehničar za elektroniku. Sve razrede prolazi s odličnim uspjehom i tokom školske godine 2008./2009. sudjeluje na državnom natjecanju iz osnova elektrotehnike. Maturirao iz navedene škole 2011.g. i te se iste godine se upisuje Elektrotehnički fakultet Osijek u sklopu Sveučilišta Josipa Jurja Štrosmajera u Osijeku, smjer elektrotehnika, te svaku godinu redovno završava. 2014. godine završava preddiplomski studij na navedenom fakultetu za smjer elektrotehnika – komunikacije i informatika, te upisuje diplomski studijski program. Na drugoj godini diplomskog studija potpisuje ugovor o stipendiranju s institutom RT-RK.

Mario Hrbak

Prilozi

Prilog 1 - dobiveni XML dokument

Prilog 1 se može CD-u pod imenom org org.hbbtv_PTR00003_original.xml

Prilog 2 - ispunjeni XML dokument

Prilog 2 se može CD-u pod imenom org.hbbtv_PTR00003.xml

Prilog 3 - html5 test dokument

Prilog 3 se može CD-u pod imenom PTR00003.html5

Prilog 4 - test plan JSON dokument

Prilog 4 se može CD-u pod imenom PTR00003.json

Prilog 5 - test plan result JSON dokument

Prilog 5 se može CD-u pod imenom PTR00003.json.result.json