



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: Немања Игњатов
Број индекса: е13048

Тема рада: Реализација комуникационог слоја ТР-069 клијента за СТВ уређај

Ментор рада: др Иштван Пап

Нови Сад, јул 2013.



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:	Немања Игњатов		
Ментор, МН:	Др Иштван Пап		
Наслов рада, НР:	Реализација комуникационог слоја TR-069 клијента за СТБ уређај		
Језик публикације, ЈП:	Српски / латиница		
Језик извода, ЈИ:	Српски		
Земља публиковања, ЗП:	Република Србија		
Уже географско подручје, УГП:	Војводина		
Година, ГО:	2013.		
Издавач, ИЗ:	Ауторски репринт		
Место и адреса, МА:	Нови Сад; трг Доситеја Обрадовића 6		
Физички опис рада, ФО: (поглавља/страница/цитата/табела/слика/графика/прилога)			
Научна област, НО:	Електротехника и рачунарство		
Научна дисциплина, НД:	Рачунарска техника		
Предметна одредница/Кључне речи, ПО:	Дигитална телевизија,Интернет,Линукс,Андроид.		
УДК			
Чува се, ЧУ:	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН:			
Извод, ИЗ:	У раду је описана библиотека за извршавање комуникације између клијента и сервера по TR-069 комуникационом протоколу, као и посебна апликација за тестирање поједињих модула TR-069 клијента.		
Датум прихватања теме, ДП:			
Датум одбране, ДО:			
Чланови комисије, КО:	Председник:	Др Милан Ђелица	
	Члан:	Др Никола Теслић	Потпис ментора
	Члан, ментор:	Др Иштван Пап	



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:	Nemanja Ignjatov		
Mentor, MN:	Ištvan Pap, PhD		
Title, TI:	Implementation of a communication layer for TR-069 client application for set-top box devices.		
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:	2013.		
Publisher, PB:	Author's reprint		
Publication place, PP:	Novi Sad, Dositeja Obradovica sq. 6		
Physical description, PD: <small>(chapters/pages/ref./tables/pictures/graphs/appendices)</small>	Digital television, Internet, Linux, Android		
Scientific field, SF:	Electrical Engineering		
Scientific discipline, SD:	Computer Engineering, Engineering of Computer Based Systems		
Subject/Key words, S/KW:			
UC			
Holding data, HD:	The Library of Faculty of Technical Sciences, Novi Sad, Serbia		
Note, N:			
Abstract, AB:	<p>This paper describes an implementation of a library used for the communication between client and server according to TR-069 communication protocol as well as separate application implemented for testing TR-069 client software modules.</p>		
Accepted by the Scientific Board on, ASB:			
Defended on, DE:			
Defended Board, DB:	President:	Milan Bjelica, PhD	
	Member:	Nikola Teslić, PhD	Menthor's sign
	Member, Mentor:	Ištvan Pap, PhD	

Zahvalnost

Za nesebičnu posvećenost i pomoć prilikom izrade završnog (Bachelor) rada, kao i brojnim savetima zahvaljujem se mentoru, doc. dr Ištvanu Papu i doc. dr Milanu Bjelici.

Takođe se zahvaljujem porodici i kolegama za podršku tokom izrade ovog rada, kao i tokom osnovnih akademskih studija.



УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА



SADRŽAJ

1.	Uvod	1
2.	Teorijske osnove.....	3
2.1	TR-069 komunikacioni protokol.....	3
2.2	Modeli podataka podržani TR-069 standardom	6
2.3	Komunikaciona sesija TR-069 protokola	8
3.	Koncept rešenja	11
3.1	Arhitektura visokog nivoa.....	12
3.1.1	Sistemski modul	13
3.1.2	TR-069 komunikacioni modul	14
3.1.3	Modul za RPC metode.....	16
3.1.4	Modul za rukovanje repozitorijumima	17
3.1.5	Ostali moduli	18
4.	Programsko rešenje.....	19
4.1	Sistemski modul.....	19
4.1.1	libcpe.h:	19
4.1.2	libcpe_config_handling.h	21
4.2	TR-069 komunikacioni modul	21
4.2.1	libcpe_engine.h.....	21
4.2.2	libcpe_comm.h	22
4.2.3	libcpe_int.h	22
4.2.4	libcpe_on_change.h	24
4.2.5	libcpe_packers.h	25
4.2.6	libcpe_parsers.h	25

4.2.7	libcpe_sched_file_transfer.h.....	26
4.2.8	libcpe_sched_trigger.h.....	27
4.2.9	libcpe_session_retry.h	28
4.2.10	libcpe_soap_fault.h.....	28
4.2.11	libcpe_soap_handling.h	28
4.3	Modul za RPC metode	29
4.4	Modul za rukovanje repozitorijumima.....	29
4.4.1	libcpe_data_repository_engine.h	29
5.	Rezultati.....	35
5.1	Test oprerećenja	35
5.2	Testiranje modula.....	36
5.3	Testiranje komunikacionog modula.....	37
6.	Zaključak	39
7.	Literatura	40

SPISAK SLIKA

Slika 1. Primer okruženja TR-069 protokola.....	4
Slika 2. Primer komunikacione sesije zasnovane na TR-069 protokolu.	10
Slika 3. Pozicioniranje libcpe u organizaciji sistema	11
Slika 4. Moduli u libcpe biblioteci.....	13
Slika 5. Mašina stanja na najvišem nivou.....	14
Slika 6. Mašina stanja za izvođenje TR-069 sesije.....	15
Slika 7. Prikaz promene vrednosti parametara kvaliteta signala u Insight projektu.....	35
Slika 8. Potrošnja memorije TR-069 klijenta tokom izvođenja testova opterećenja.....	36
Slika 9. Izgled aplikacije za testiranje modula TR-069 klijenta	36
Slika 10. Rezultati izvršavanja testa za modul za rukovanje repozitorijumima	37
Slika 11. Prikaz rezultata testiranja TR-069 klijenta korišćenjem InsightACS.....	38

SPISAK TABELA

Tabela 1. Protokol stek TR-069 protokol standarda	4
Tabela 2. Zahtevi za podrškom RPC metoda po TR-069 verzijama standarda	5
Tabela 3. Modeli podataka podržani TR-069 protokolom	6
Tabela 4. Tipovi događaja u TR-069 standardu	9
Tabela 5. Opisi funkcija za kreiranje HTTP zahteva	25
Tabela 6. Opisi funkcija za parsiranje SOAP struktura	25
Tabela 7. Opisi funkcija za rukovanje SOAP strukturama	29

SKRAĆENICE

CPE	- Customer Premises Equipment - Krajnji korisnički uređaj
ACS	- Auto Configuration Server - Poslužilac za automatsku konfiguraciju
SOAP	- Simple Object Access Protocol – Komunikacioni protokol za razmenu objekata podataka
SNMP	- Simple Network Management Protocol – Protokol za upravljanje uređajima u IP mrežama
XML	- Extensible Markup Language – Standardni skup pravila za definisanje formata podataka u elektronskoj formi.
HTTP	- HyperText Transfer Protocol – Protokol za prenos hiperteksta
RPC	- Remote Procedure Call – poziv udaljene metode
FIFO	- First In First Out – struktura podataka tipa red
STB	- Set-Top Box – digitalni TV prijemnik
CRUD	- Create Read Update Delete – operacije nad strukturama podataka
VoIP	- Voice over Internet Protocol – Metodologija prenosa glasa preko internet protokola
IPTV	- Internet Protocol TeleVision – sistem prenosa televizijskog signal putem interneta
NAS	- Network-Attached Storage – Memorija za čuvanje podataka na mreži
FAP	- Femtocell Access Point - Pristupna tačka zasnovana na baznoj stanici niske potrošnje
OS	- Operative System – operativni sistem
OSAL	- Operative System Abstraction Layer – prilagodni sloj operativnog sistema
DEVAL	- DEVice Abstraction Layer – prilagodni sloj uređaja
API	- Application Programming Interface – sprega za razvoj programske podrške

1. Uvod

Sa ubrzanim razvojem Internet tehnologija došlo je do sve veće potrebe za praćenjem parametara krajnih uređaja u mrežama, automatskim konfiguriranjem istih, udaljenom podrškom korisnicima, kao i mogućnošću automatskog ažuriranja programske podrške na velikom broju uređaja. Kao odgovor na ove zahteve dizajniran je SNMP (eng. Simple network management protocol) protokol krajem prošlog veka, koji je imao brojne prednosti, ali nije pružio dovoljno standardizovanu podršku za uređaje različitih proizvođača, što je kasnije prouzrokovalo brojne probleme sa sigurnošću sistema. Kao rešenje za to je osmišljen TR-069 protokol koji je nezavistan od tipa uređaja i proizvođača. Dizajniran prvenstveno za prikupljanje podataka o krajnjim uređajima u mreži, kao i njihovu konfiguraciju, TR-069 protokol se zasniva na vezi za siguran prenos podataka, gde se ne zahteva velika brzina odziva. Najvažniji protokol na koji se oslanja TR-069 je HTTP/HTTPS, koji obezbeđuje siguran transfer podataka. Isto tako, korišćenjem HTTP protokola, prenos podataka je uvek iniciran od strane HTTP klijenta, u ovom slučaju i TR-069 klijenta, tako da nije potrebno održavanje sesije na strani poslužioca. Iz istog razloga, opsluživanje velikog broja TR-069 klijenata od strane poslužioca ne uslovjava dodatni utrošak sistemskih resursa.

U ovom radu je realizovan klijent zasnovan na TR-069 komunikacionom protokolu, kao i skup proširenja radi prilagođavanja rada klijenta na ciljnoj STB (set-top box) platformi. Akcenat je stavljen na laku prenosivost programske podrške između raznih platformi, kao i nezavisnost od tipa uređaja koji koristi TR-069 klijent. Rešenje je implementirano na operativnom sistemu Linuks u programskom jeziku C i prenešeno na operativne sisteme Android i STLinuks. Realizacija klijenta je organizovana u više modula radi lakšeg razvoja i održavanja programske podrške. Funkcionalnosti klijenta su integrisane i testirane sa InsightACS v1.0[5], InsightACS v0.5 i OpenACS[4].

Ovaj rad je sačinjen od sedam poglavlja.

Drugo poglavlje izlaže teorijske osnove, pre svega TR-069 protokola i njemu pridruženih modela podataka za različite tipove uređaja.

U trećem poglavlju dat je koncept rešenja za implementaciju TR-069 klijenta, kao i njegovih prilagodnih slojeva za STB uređaje.

Četvrto poglavlje daje detaljan opis modula unutar biblioteke i načine njihovog sprezanja.

U petom poglavlju su dati načini ispitivanja i verifikovanja funkcionalnosti TR-069 klijenta.

Šesto poglavlje sadrži kratak pregled urađenog i moguće dalje pravce razvoja.

U sedmom poglavlju navedena je korišćena literatura.

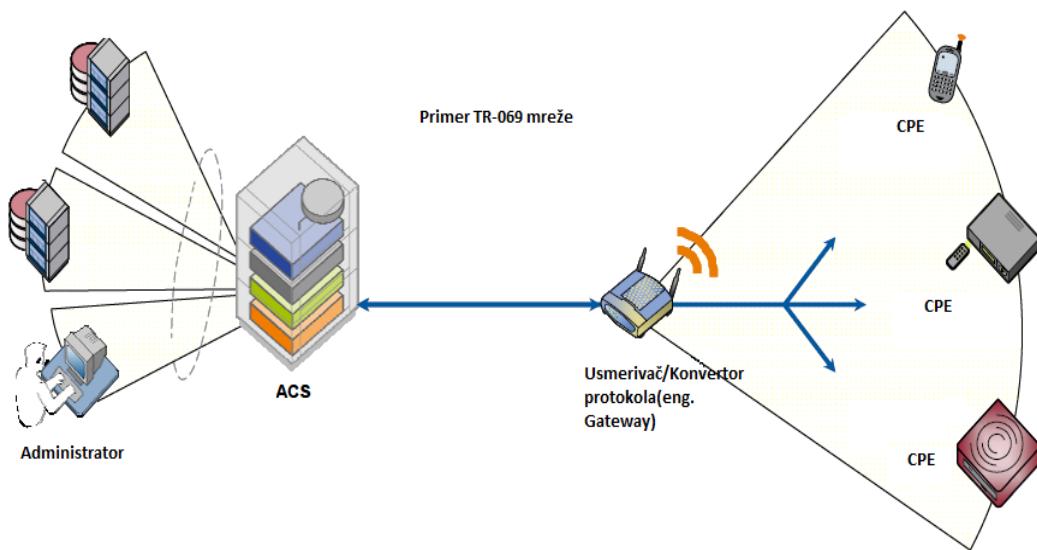
2. Teorijske osnove

U ovom poglavlju date su teorijske osnove neophodne za implementaciju biblioteke koja omogućava funkcionalnosti klijenta zasnovanog na TR-069 protokolu. Biblioteka predstavlja spregu između programske podrške krajnjih uređaja i poslužioca za automatsku konfiguraciju (ACS. Auto-Configuration Server) i jedan je od delova sistema korišćenog za nadgledanje vrednosti parametara kvaliteta televizijskog signala.

2.1 TR-069 komunikacioni protokol

TR-069 (eng.Technical Report 069) definiše protokol na aplikativnom nivou protokol steka za udaljeno rukovanje krajnjim uređajima[1]. Korišćenjem ovog protokola veza se uvek uspostavlja između krajnjeg uređaja (CPE. Customer Premises Equipment) i ACS-a. Protokol se zasniva se na bidirekcionoj SOAP/HTTP vezi. Protokolom je opisan samo način prenosa podataka između navedenih struktura u mreži, dok su definicije podataka koji se prenose date u modelima podataka pridruženih TR-069 standardu. U ovom radu korišćen je TR-135 model podataka[2], namenjen STB uređajima, kao i TR-106[3] koji definiše osnovne podatke sistema potrebne za uspostavu TR-069 veze. Činjenice da se zasniva na HTTP protokolu i da je nezavistan od tipa uređaja donele su TR-069 protokolu široko polje primene. Neke od osnovnih funkcionalnosti koje pruža TR-069 standard su:

- Automatska konfiguracija i dinamičko omogućavanje usluga
- Rukovanje programskom podrškom krajnjeg uređaja
- Nagledanje statusa i performansi krajnjeg uređaja i njegova dijagnostika



Slika 1. Primer okruženja TR-069 protokola

Kao što se može videti na slici 1. CPE uređaji mogu biti različitog tipa, a pritom ne postoje razlike u protokolu po kom se prenose podaci do ACS-a, razlikuju se samo modeli podataka za svaki od datih CPE-ova. TR-069 standard definiše niz protokola koji se moraju koristiti i na CPE i na ACS strani prenosa podataka.

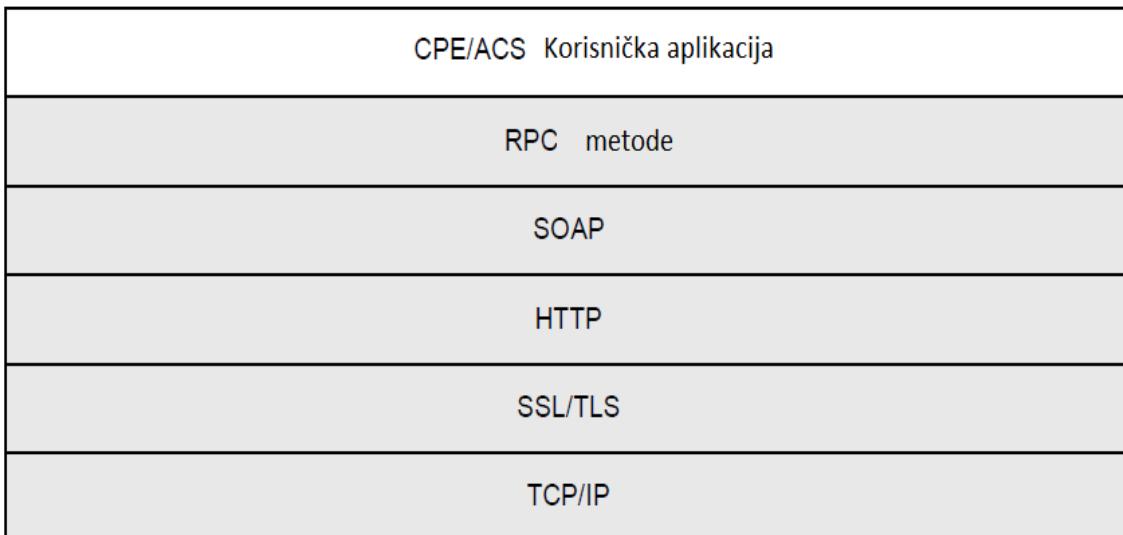


Tabela 1. Protokol stek TR-069 protokol standarda

Kao što se može videti u tabeli 1. osnovni cilj TR-069 protokola je bezbedan prenos podataka, dok brzina prenosa i kašnjenje nije toliko bitno. Velika prednost je i što je obavezno korišćenje SSL i TLS enkripcionih algoritama, koji se trenutno najviše koriste u računarskim mrežama radi očuvanja sigurnosti prenosa informacija.

Poruke koje se prenose putem TR-069 protokola su standardizovane, gde je za svaku od njih definisana sintaksa poruke, način rukovanja, kao i rukovanje u slučaju greške. Prilikom standardizacije je korišćen SOAP protokol, koji koristi XML format.

Upravljačke poruke korišćene u TR-069 protokolu su definisane kao RPC metode, čime je omogućena jednostavna proširivost standarda. Ukoliko je potrebno dodavanje nove upravljačke poruke, potrebno je samo definisati novu RPC metodu, način rukovanja i njenu sintaksu na SOAP nivou, bez potrebe za menjanjem nižih nivoa protokol steka.

RPC metoda	TR-069 amendment 1	TR-069 amendment 2	TR-069 amendment 3	TR-069 amendment 4
GetRPCMethods	Obavezna	Obavezna	Obavezna	Obavezna
SetParameterValues	Obavezna	Obavezna	Obavezna	Obavezna
GetParameterValues	Obavezna	Obavezna	Obavezna	Obavezna
GetParameterNames	Obavezna	Obavezna	Obavezna	Obavezna
SetParameterAttributes	Obavezna	Obavezna	Obavezna	Obavezna
GetParameterAttributes	Obavezna	Obavezna	Obavezna	Obavezna
AddObject	Obavezna	Obavezna	Obavezna	Obavezna
DeleteObject	Obavezna	Obavezna	Obavezna	Obavezna
Download	Obavezna	Obavezna	Obavezna	Obavezna
Reboot	Obavezna	Obavezna	Obavezna	Obavezna
Inform	Obavezna	Obavezna	Obavezna	Obavezna
TransferComplete	Obavezna	Obavezna	Obavezna	Obavezna
Upload	Opciona	Opciona	Opciona	Opciona
FactoryReset	Opciona	Opciona	Opciona	Opciona
GetQueuedTransfers	Opciona	Opciona	Zastarela	Zastarela
ScheduleInform	Opciona	Opciona	Opciona	Opciona
SetVouchers	Opciona	Opciona	Zastarela	Zastarela
GetOptions	Opciona	Opciona	Zastarela	Zastarela
GetRPCMethods (ACS)	Opciona	Opciona	Opciona	Opciona
RequestDownload	Opciona	Opciona	Opciona	Opciona
Kicked	Opciona	Opciona	Zastarela	Zastarela
GetAllQueuedTransfers		Opciona	Opciona	Opciona
AutonomousTransferComplete		Opciona	Opciona	Opciona
DUStateChangeComplete			Opciona	Opciona
AutonomousDUStateChangeComplete			Opciona	Opciona
CancelTransfer			Opciona	Opciona
ScheduleDownload			Opciona	Opciona
ChangeDUState			Opciona	Opciona

Tabela 2. Zahtevi za podrškom RPC metoda po TR-069 verzijama standarda

Kao što se može videti u tabeli 2. sa svakom verzijom standarda dolazi do dodavanja novih i izbacivanja zastarelih RPC metoda. Međutim, jedan od osnovnih zahteva TR-069 protokola je kompatibilnost sa prethodnim verzijama standarda, stoga je potrebno implementirati sve prethodne verzije standarda, bez obzira koju verziju koristimo.

2.2 Modeli podataka podržani TR-069 standardom

Kao što je već rečeno, široko polje primeni TR-069 standarda je omogućila primenljivost na različite tipove uređaja bez potrebe za menjanjem načina prenosa poruka definisanih protokolom. Neki od modela podataka se navedeni u tabeli 3:

Model podataka	Tip uređaja
TR-098	Internet konvertori prokola i usmerivači
TR-104	VoIP uređaji
TR-106	Osnovni model podataka za TR-069
TR-135	IPTV i STB uređaji
TR-140	NAS uređaji
TR-143	Model korišćen za dijagnostiku CPE
TR-157	Model komponenata datog CPE
TR-192	FAP uređaji

Tabela 3. Modeli podataka podržani TR-069 protokolom

Modeli podataka, definisani TR-106 modelom podataka[3], su strukturirani kao stablo, u kom postoje sledeći tipovi čvorova:

- Korenski čvor
- Objekat
- Višestruki objekat
- Parametar

Korenski čvor omogućava objedinjavanje više modela podataka u jedno stablo ukoliko je to potrebno za određeni uređaj. Objekat predstavlja čvor u stablu koji ne može biti list, već se samo koristi za bolje struktuiranje podataka. Višestruki objekat je takođe objekat, s tim što se može više puta instancirati i pritom mu se pridružuje indeks pomoću kog mu se može jednoznačno pristupiti. Parametri predstavljaju listove u stablu i samo njima je pridružena vrednost. Pored vrednosti parametrima se pridružuje i niz atributa pomoću kojih se vrši konfiguracija uređaja od strane ACS-a. Atributi su:

- Nivo notifikacije (eng. Notification) , koji definiše od kojim okolnostima CPE treba da javi promenu vrednost ACS-u.
- Lista pristupa (eng. AccessList), definiše listu entiteta koji smeju da promene vrednost parametra.

Pored atributa, modelom se definišu i podešavanja koja se ne mogu promeniti od strane ACS-a. Podešavanja su:

- Mogućnost upisa vrednosti parametra
- Ograničenja vrednosti parametra, definisana enumeracijom ili graničnim vrednostima parametara
- Obaveza slanja vrednosti parametra prilikom iniciranja TR-069 sesije
- Tip podatka datog parametra
- Mogućnost nadgledanja promene vrednosti parametra u realnom vremenu
- Početna podrazumevana vrednost parametra
- Ograničenje broja višestruktih čvorova kod istog pretka.
- Da li je čvor definisan u tom profilu modela podataka
- Jedinica mere date vrednosti parametra

Tipovi podataka parametara koje definiše standard su:

- Niz karaktera (eng. String)
- Celobrojni (eng. Integer)
- Celobrojni prošireni (eng. Long)
- Neoznačeni celobrojni (eng. Unsigned Integer)
- Neoznačeni prošireni celobrojni (eng. Unsigned Long)
- *Boolean*
- Vremenski tip (eng. dateTime)
- Binarni tip (eng. Base64)
- Heksadecimalni tip (eng. hexBinary)

TR-069 strandard definiše i tipove operacija koje se mogu izvršiti nad stablom:

- Dodavanja i brisanje čvora je moguće samo ukoliko je dati čvor višestruki objekat.
- Dobavljanje vrednosti je moguće samo ukoliko je čvor parametar
- Postavljanje vrednosti je moguće ukoliko je čvor parametar i moguća je promena njegove vrednosti
 - Postavljanje vrednosti atributa nad parametrom

Takođe se uz svaki čvor pridružuje semantički opis tog čvora definisan stringom razumljivom čoveku.

Važno je naglasiti da se prilikom pristupa bilo kom čvoru u stablu mora navoditi njegova cela putanja od korenskog čvora, gde se kao separator između imena čvorova navodi karakter tačka. Primer putanje:

KorenskiČvor.Objekat.VišestrukiObjekat.Indeks.Objekat.Parametar

2.3 Komunikaciona sesija TR-069 protokola

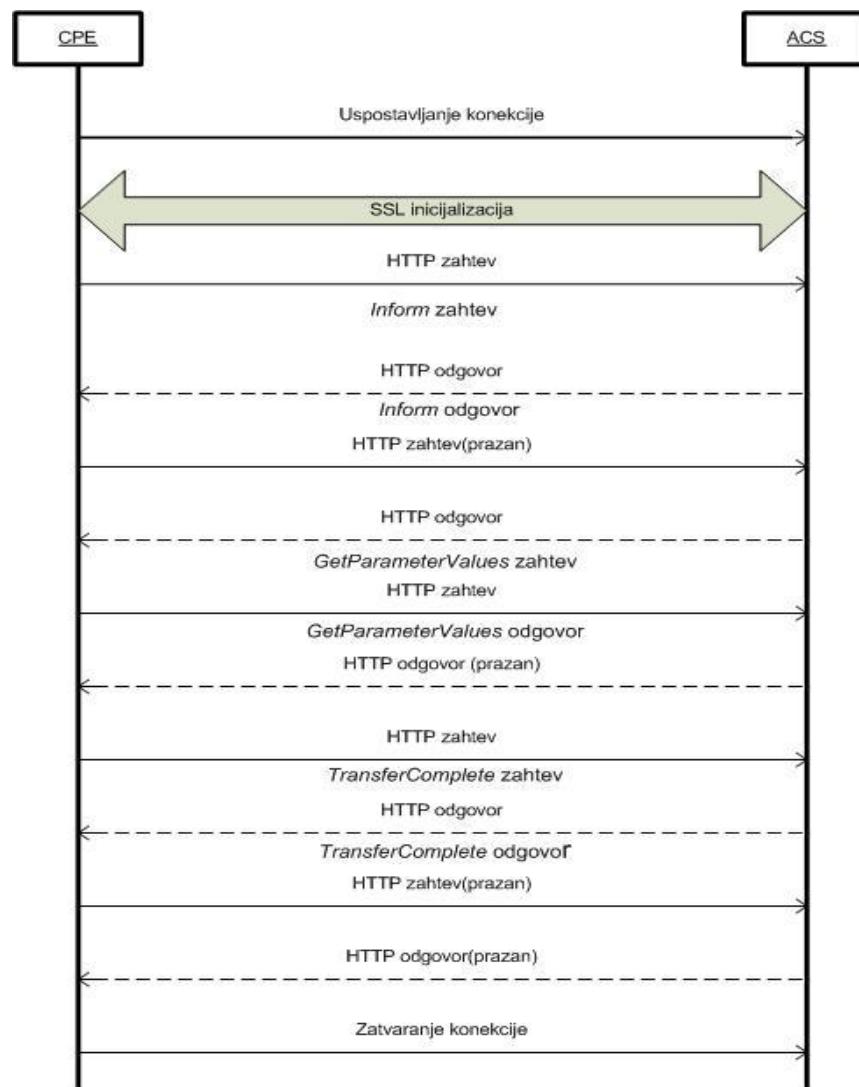
Komunikaciona sesija definisana TR-069 standardom se uvek uspostavlja između jednog CPE-a i ACS-a. U jednom trenutku CPE sme biti u samo jednoj sesiji sa samo jednom ACS-om. CPE je HTTP klijent i uvek šalje HTTP zahteve, dok ACS uvek šalje HTTP odgovore. Pokretanje sesije mora biti pobuđeno određenim događajem koji se desio na CPE strani. CPE je obavezan da informiše ACS o događajima koji su se desili i zbog kojih je pokrenuta sesija u prvoj poruci u sesiji, *Inform* zahtevu. Događaji definisani TR-069 standardom nalaze se u tabeli 4:

Vrsta događaja	Značenje
0 Bootstrap	Prilikom prvog uključivanja CPE-a i prijavljanja kod ACS-a.
1 Boot	Prilikom pokretanja CPE-a.
2 Periodic	Prilikom isteka predefinisanog intervala
3 Scheduled	Ako je ACS zakazao pokretanje sesije za određeno vreme
4 Value Change	Ukoliko se desila promena vrednosti parametra koji ACS želi da nagleda
5 Kicked	Ukoliko je došlo do promene internet stranice na strani CPE
6 Connection request	Ukoliko je ACS zatražio pokretanje sesije
7 Transfer Complete	Ukoliko se završio transfer fajla od strane CPE-a
8 Diagnostics Complete	Posle završetka dijagnostike na CPE-u
9 Request Download	Ukoliko CPE proverava da li postoji fajl koji treba da preuzme
M Reboot	Ukoliko se desilo ponovno pokretanje uređaja po zahtevu ACS-a
M ScheduleInform	Ako je ACS zakazao pokretanje sesije za određeno vreme više puta
M Download	Ukoliko je završeno preuzimanje fajla
M Upload	Ukoliko je završeno postavljanje fajla na određeni poslužilac
M <vendor-specific method>	Ukoliko je izvršena određena metoda

	specifična za dati uređaj
X <OUI> <event>	Ukoliko se desio događaj specifičan za dati uređaj

Tabela 4. Tipovi događaja u TR-069 standardu

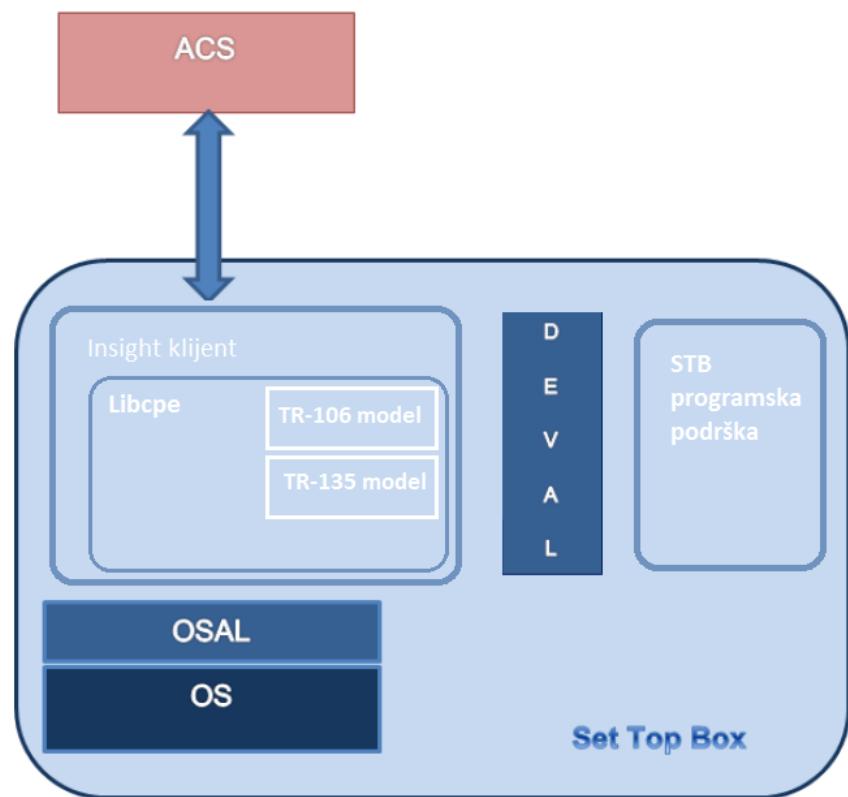
Posle pokretanja sesije CPE i ACS šalju jedan drugom SOAP zahteve i odgovore, s tim što u sesiji u jednom trenutku ne može postojati više od jednog SOAP zahteva na koji nije odgovoren. Da bi se omogućila bidirekciono slanje zahteva, CPE i ACS tokom sesije postaju SOAP pošiljalac (eng. SOAP sender) i SOAP primalac (eng. SOAP receiver). ACS prvenstveno ima pravo da šalje SOAP zahteve, a mogućnost da šalje SOAP zahteve može dati CPE-u tako što pošalje prazan HTTP odgovor ili postavi kontrolnu promenljivu za slanje zahteva (eng. HoldRequests) na 0. Nakon obrade svih zahteva na ACS i CPE strani sesija je zatvara slanjem 2 prazna HTTP paketa. Ukoliko u bilo kom trenutku tokom sesije dođe do greške koja se ne odnosi na rukovanje RPC metodama, CPE je dužan da sesiju prekine odmah i ne pošalje nijedan HTTP zahtev više, a da nasilno prekinutu sesiju ponovno započne posle određenog intervala vremena ili ako se desi neki drugi događaj o kom treba obavestiti ACS. Ukoliko se desi greška prilikom rukovanja RPC metodom, umesto SOAP odgovora poslaće se SOAP izveštaj o grešci (eng. SOAP fault) i sesija će biti nastavljena. Primer TR-069 komunikacione sesije je dat na slici 2.



Slika 2. Primer komunikacione sesije zasnovane na TR-069 protokolu.

3. Koncept rešenja

Biblioteka (u daljem tekstu libcpe) koja implementira funkcionalnosti TR-069 klijenta se koristi kao deo sistema koji obuhvata prikupljanje podataka sa krajnjih uređaja, njihovo skladištenje, kao i prosleđivanje podataka ACS-u. U takvom sistemu biblioteka predstavlja spregu između krajnjeg uređaja, u konkretnom slučaju STB-a i ACS-a.



Slika 3. Pozicioniranje libcpe u organizaciji sistema

Kao što se može videti sa slike 3. Libcpe predstavlja deo programske podrške uređaja koja svoj rad zasniva na prilagodnim slojevima prema:

- operativnom sistemu (OSAL – Operative system abstraction layer)
- krajnjem uređaju (DEVAL – Device abstraction layer)

Korišćenjem prilagodnog sloja prema operativnom sistemu obezbeđene su funkcionalnosti za upotrebu sistemskog sata, mrežnih sprega, kreiranje niti i njihovu sinhronizaciju, kreiranje kritičnih sekcija, zauzimanje i oslobađanje memorije.

Korišćenjem prilagodnog sloja prema uređaju obezbeđene su funkcionalnosti za dobavljanje i postavljanje vrednosti parametara, dijagnostiku rada, kao i za ažuriranje programske podrške uređaja.

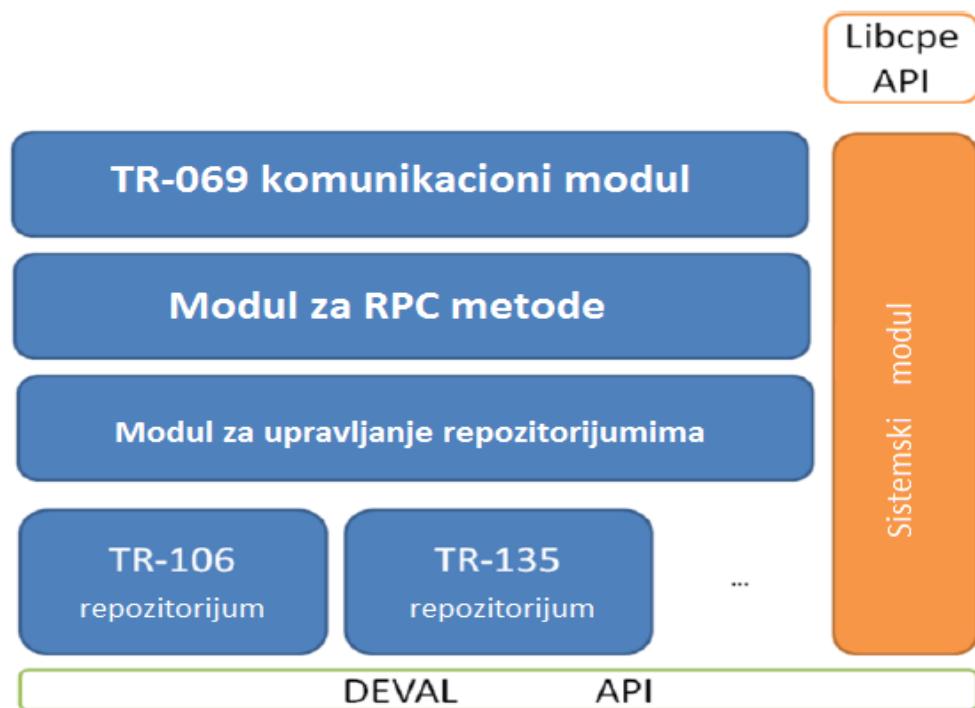
STB programska podrška predstavlja skup funkcionalnosti koje obezbeđuju rad modernih digitalnih televizijskih prijemnika. Neki od njih su :

- Prijem signala
- Reprodukciju slike i zvuka
- Snimanje multimedijalnih sadržaja
- Prikaz programske vodiča i teleteksta

Libcpe predstavlja celinu u programskoj podršci uređaja koji koji na jednoj strani omogućava komunikaciju zasnovanu na TR-069 protokolu, dok na drugoj razmenjuje podatke sa STB programskom podrškom. Funkcionalnosti libcpe-a su enkapsulirane u Insight klijenta, koji još sadrži module zasnovane na TR-106 i TR-135 modelima podataka, radi lakšeg prilagođavanja STB programskoj podršci.

3.1 Arhitektura visokog nivoa

Jedan od osnovnih ciljeva pri dizajniranju biblioteke je modularnost i jednostavna proširivost, da bi održavanje koda bilo što lakše i kasnije proširivanje sa novijim TR-069 standardima, pre svega novim RPC metodama. Poseban aspekt je i organizacija repozitorijuma biblioteke, tako da se na lak način mogu proširiti novim modelima podataka podržanih u TR-069 standardu, bez promena u samom kodu biblioteke, već samo definisanjem i dodavanjem novih XML datoteka. Organizacija modula u okviru biblioteke je data na slici 4.



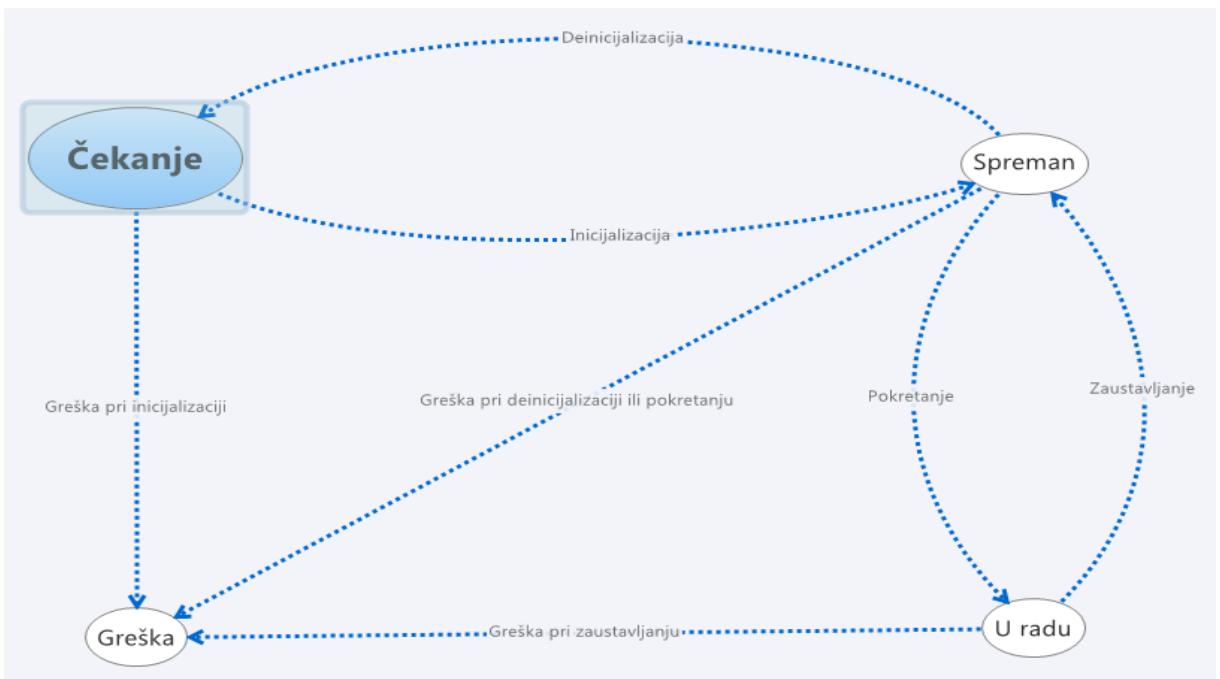
Slika 4. Moduli u libcpe biblioteci

3.1.1 Sistemska modul

Sistemska modul predstavlja kontrola operacija nad celom bibliotekom:

- Inicijalizuje sve ostale module
- Pokreće i zaustavlja rad TR-069 klijenta
- Omogućava dostupnost funkcionalnosti biblioteke drugim aplikacijama

U sklopu sistemskog modula je omogućeno pokretanje i zaustavljanje rada biblioteke pozivom odgovarajućih funkcija. Prilikom pokretanja biblioteke ona se nalazi u stanju čekanja i pozivom funkcije za inicijalizaciju prelazi u stanje spreman. Pri inicijalizaciji se zauzima potrebna memorija i instanciraju moduli potrebni za komunikaciju sa poslužiocem, kao i za spregu sa programskom podrškom krajnjeg uređaja. Kada se nalazi u stanju spreman moguće je pokrenuti TR-069 komunikacioni modul i tada počinje nadgledanje događaja na krajnjem uređaju i komunikacija sa ACS-om. Kada se biblioteka nađe u stanju "u radu", moguće je zaustaviti njen rad i tada prelazi u stanje spreman. Važno je naglasiti da zaustavljanje rada može da traje izvesno vreme pošto je zbog očuvanja konzistentnosti podataka potrebno sačekati završetak tekuće TR-069 sesije. Po završetku rada sa bibliotekom potrebno je pozvati deinicijalizaciju radi oslobađanja zauzete memorije. Ukoliko dođe do greške u radu biblioteke koja se ne odnosi na tok TR-069 sesije prelazi se u stanje greške. Prelasci između stanja na najvišem nivou su prikazani na slici 5.

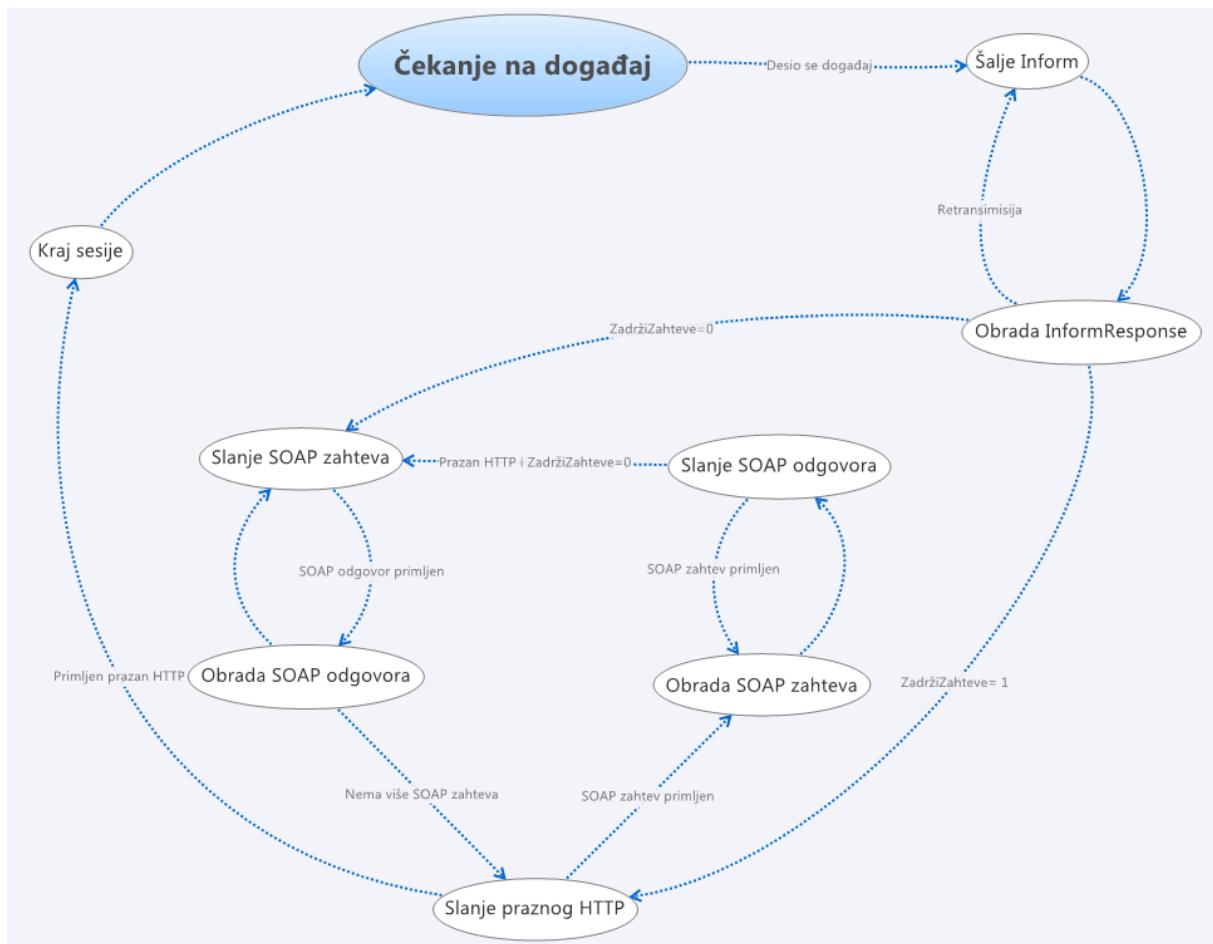


Slika 5. Mašina stanja na najvišem nivou

3.1.2 TR-069 komunikacioni modul

TR-069 komunikacioni modul implementira deo TR-069 standarda koji se odnosi na nadgledanje događaja na krajnjem uređaju i izvođenje komunikacione sesije. Jedino ovaj modul ima direktnu dostupnost prema ACS-u. U sklopu modula su funkcionalnosti:

- Čekanje i obradu događaja koji će pokrenuti TR-069 sesiju
- Izvođenje TR-069 sesije
- Slanje i prijem HTTP poruka
- Parsiranje HTTP zaglavlja i proveru ispravnosti
- Pakovanje, validaciju i parsiranje SOAP struktura
- Pozivanje odgovarajuće RPC metode
- Izvođenje prenosa datoteka
- Omogućavanje sprege za rukovanje programskom podrškom krajnjeg uređaja (eng. Firmware upgrade)



Slika 6. Mašina stanja za izvođenje TR-069 sesije

Suštinu modula predstavlja mašina stanja pomoću koje se izvodi TR-069 sesija. Mašina stanja može da bude pokrenuta samo ako se klijent nalazi u stanju “u radu”. Mašina čeka na događaj na krajnjem uređaju i ukoliko se desio barem jedan kreće u komunikaciju sa ACS-om. Prolasci kroz stanja su u skladu sa TR-069 standardom. Kao što se može videti sa slike 6. najvažnije za dalji tok sesije je :

- Da li je ACS poslao prazan HTTP paket
- Vrednost kontrolne promenljive ZadržiZahteve (eng. HoldRequests) u poslednjem SOAP zahtevu od strane ACS.

Inicijalno ACS ima prvenstvo da šalje SOAP zahteve, međutim ako CPE obradi i odgovori na sve potrebne zahteve, ACS prepušta prvenstvo slanja SOAP zahteva CPE-u postavljanjem vrednosti promenljive ZadržiZahteve na nulu u poslednjem SOAP zahtevu koji šalje i slanjem praznog HTTP paketa posle primljenog SOAP odgovora.

Prilikom prijema svakog HTTP paketa vrši se parsiranje HTTP zaglavila, a potom provera ispravnosti SOAP paketa. Ukoliko je primljen SOAP zahtev određuje je koji tip RPC metode se poziva i vrši se parsiranje SOAP paketa i poziv RPC metode sa dobijenim podacima. Ukoliko je poziv RPC metode prošao bez grešaka šalje se odgovarajući SOAP

odgovor, u drugom slučaju umesto SOAP odgovora šalje se SOAP izveštaj o grešci (eng. SOAP Fault). Ukoliko je primljen SOAP odgovor CPE mora da ažurira odgovarajuće redove poruka. Ukoliko je primljen SOAP izveštaj o grešci, u zavisnosti od primljenog koda greške CPE je dužan da pokuša ponovno slanje poslednje poslatog SOAP zahteva, ili da odloži ponovno slanje datog zahteva za sledeću sesiju. U slučaju da se desi greška pri proveri ispravnosti HTTP zaglavla ili SOAP struktura, CPE mora nasilno da prekine sesiju.

3.1.3 Modul za RPC metode

Modul za RPC metode predstavlja proširenje TR-069 komunikacionog modula, međutim zbog lakšeg održavanja i jednostavnog proširivanja biblioteke izmešten je kao zaseban modul. Razlog za to je razlike između različitih profila TR-069 standarda. Naime, samo jezgro za komunikaciju je slabo promeljivo u profilima standarda, dok se najveće promene odnose na RPC metode, dodavanje novih i proglašavanje pojedinih za zastarele. Uz svaku RPC metodu su realizovane tri osnovne funkcije:

- Funkcija za otpakivanje SOAP zahteva ili SOAP odgovora
- Funkcija za obradu određene RPC metode
- Funkcija za pakovanje SOAP zahteva ili SOAP odgovora

Sve ostale funkcije u modulu su specifične za određenu RPC metodu.

U zavisnosti od vrste RPC metode poziva se odgovarajuća obrada. U odnosu na tip obrade koja je zahtevana RPC metode možemo podeliti na 3 grupe:

- Operacije nad repozitorijumima
 - o *AddObject* – dodavanje višestrukog objekta u repozitorijum
 - o *DeleteObject* – uklanjanje višestruktor objekta iz repozitorijuma
 - o *SetParameterValues* – postavljanje vrednosti parametara
 - o *GetParameterValues* – dobavljanje vrednosti parametara
 - o *SetParameterAttributes* – postavljanje vrednosti atributa određenog parametra
 - o *GetParameterAttributes* – dobavljanje vrednosti atributa određenog parametra
 - o *GetParameterNames* – dobijanje putanja čvorova određenog podstabla
- Prenose datoteka
 - o *Download* – preuzimanje određene datoteke sa poslužiocu
 - o *Upload* – postavljanje određene datoteke na poslužilac
 - o *TransferComplete* – izveštaj o prenosu datoteke

- *RequestDownload* – provera da li postoji fajl za preuzimanje na poslužiocu
- *GetQueuedTransfers* – dobavljanje svih prenosa datoteka koji čekaju na izvršenje
- Upravljanje krajnjim uređajem
 - *Reboot* – ponovno pokretanje uređaja
 - *FactoryReset* – vraćanje uređaja na fabrička podešavanja
 - *SetVouchers* – postavljanje dozvola za izvršavanje funkcionalnosti uređaja
 - *GetOptions* – dobavljanje podržanih funkcionalnosti na uređaju
 - *Kicked* – promena internet stranice na uređaju
- Upravljanje komunikacijom
 - *Inform* – poruka za započinjanje TR-069 sesije
 - *GetRPCMethods* – dobavljanje podžanih RPC metoda
 - *ScheduleInform* – zakazivanje TR-069 sesije za određeno vreme

3.1.4 Modul za rukovanje repozitorijumima

Modul za rukovanje repozitorijumima (eng. Data Repository Engine) obezbeđuje CRUD funkcionalnosti nad repozitorijumima definisanim na osnovu TR-069 podržanih modela podataka. Uređenje struktura u ovom modulu omogućuje lako proširivanje novim modelima podataka. Svaki model podataka se prilikom inicijalizacije biblioteke instancira kao stablo i poveže u listu repozitorijuma, tako se dodavanje novih repozitorijuma može izvršiti bez izmena u kodu biblioteke. Nad svakim repozitorijumom se definiše skup od maksimalno dva pretplatnika, koji se obaveštavaju da li se desila neka od CRUD operacija nad repozitorijumom. Na taj način je od svih pretplatnika sakrivena unutrašnja organizacija repozitorijuma i onemogućena je zloupotreba repozitorijuma. Prilikom inicijalizacije biblioteke sistemski modul obavi sva potrebna instanciranja modela podataka i registracije pretplatnika, tako da je naknadno registrovanje nemoguće. Prilikom registrovanja svaki pretplatnik dobija identifikator preko kog vrši promene u repozitorijumu. Podrazumevan scenario je da se prilikom inicijalizacije TR-069 komunikacioni modul registruje kao pretplatnik na sve repozitorijume, a da drugi pretplatnik bude prilagodni sloj uređaja (eng. Device Abstraction Layer). Osnovni skup funkcionalnosti koje omogućava modul za rukovanje repozitorijumima su:

- Inicijalizacija modula
- Instanciranje repozitorijuma uz pomoć modela podataka
- Registrovanje pretplatnika na repozitorijum

-
- Odjava pretplatnika sa repozitorijuma
 - Postavljanje vrednosti parametara
 - Dobavljanje vrednosti parametara
 - Dodavanje višestrukog objekta
 - Uklanjanje višestrukog objekta

Pored ovih funkcionalnosti radi lakše integracije u biblioteku dodate su funkcije specifične za određene RPC metode. Upotreba ovih funkcija nije dozvoljena nijednom drugom osim TR-069 komunikacionom modulu. Dodatne funkcionalnosti su:

- Provera ispravnosti zahteva za postavljanjem vrednosti parametra
- Dobavljanje liste vrednost parametara
- Dobavljanje liste vrednosti atributa parametra
- Dobavljanje liste putanja čvorova u podstablu repozitorijuma
- Dobavljanje vrednosti parametara koji se moraju poslati u *Inform* zahtevu
- Postavljanje identifikatora vrednosti parametara (eng. ParameterKey)

3.1.5 Ostali moduli

Moduli *TR-106 repozitorijum* i *TR-135 repozitorijum* predstavljaju spregu prilagodnih slojeva konkretnih krajnjih uređaja prema biblioteci. Zasnovani su na odgovarajućim modelima podataka.

Libcpe API i *DEVAL API* su .h fajlovi koji se isporučuju korisnicima biblioteke i sadrže kompletan skup funkcija kojima se može iskoristiti potpuna funkcionalnost libcpe-a.

4. Programsко rešenje

U ovom poglavlju je opisano programsko rešenje biblioteke koja implementira funkcionalnosti klijenta zasnovanog na TR-069 standardu. Rešenje je realizovano u programskom jeziku C na operativnom sistemu Linuks. Važno je napomenuti zavisnosti od biblioteka:

- libxml2
- libcurl
- pthread

TR-069 klijent se zasniva na radu više niti, koje se međusobno sinhronizuju:

- Komunikaciona nit
- Nit za korišćenje maštine stanja TR-069 sesije
- Nit za periodično pokretanje TR-069 sesije
- Nit za raspoređivanje prenosa datoteka
- Nit za retransmisiju TR-069 sesije
- Nit za HTTP poslužilac

Sinhronizacija niti se izvodi FIFO strukturama podataka i uslovnim promenljivama (eng. Condition Variable).

4.1 Sistemski modul

U okviru sistemskog modula realizovane su osnovne funkcionalnosti za rukovanje kontekstom CPE klijenta, strukture podataka na najvišem nivou, koja enkapsulira sve ostale strukture podataka.

4.1.1 libcpe.h:

- cpe_err_t cpe_init(void ** const ctx_pptr);

- Opis: Inicijalizacija CPE konteksta, pritom se učitavaju podaci iz konfiguracionog fajla, kao i podaci potrebni za uspostavu veze sa ACS-om. Takođe dolazi do instanciranja potrebnih repozitorijuma i registrovanje preplatnika na te repozitorijume.
 - Parametri: ctx_pptr - Kontekst CPE klijenta koji treba inicijalizovati
 - Povratna vrednost :Indikacija uspešnosti inicijalizacije
- cpe_err_t cpe_is_valid_ctx(void *ctx_ptr);
- Opis: Provera da li je došlo do narušenja konzistentnosti CPE konteksta.
 - Parametri: ctx_ptr - Kontekst CPE klijenta koji treba proveriti
 - Povratna vrednost : Da li je kontekst narušen ili nije
- cpe_err_t cpe_destroy(void * const ctx_ptr);
- Opis: Deinicijalizacija CPE klijenta, zaustavljanje klijenta ukoliko nije prethodno zaustavljen
 - Parametri: ctx_ptr - Kontekst CPE klijenta koji treba deinicijalizovati
 - Povratna vrednost : Indikacija uspešnosti deinicijalizacije
- cpe_err_t cpe_start(void * const ctx_ptr);
- Opis: Pokretanje niti koje vrše komunikaciju sa ACS-om kao i pomoćnih niti potrebnih za rad klijenta
 - Parametri: ctx_ptr - Kontekst CPE klijenta koji treba pokrenuti
 - Povratna vrednost : Indikacija uspešnosti pokretanja
- cpe_err_t cpe_stop(void * const ctx_ptr);
- Opis: Zaustavljanje niti koje vrše komunikaciju sa ACS-om kao i pomoćnih niti potrebnih za rad klijenta
 - Parametri: ctx_ptr - Kontekst CPE klijenta koji treba zaustaviti
 - Povratna vrednost : Indikacija uspešnosti zasutavljanja
- cpe_err_t cpe_get_status(void* ctx_ptr, cpe_status_t *cpe_status);
- Opis: Dobavljanje stanja u kom se trenutno nalazi CPE klijent
 - Parametri:
 - ctx_ptr - Kontekst CPE klijenta
 - cpe_status - Status u kom se nalazi dati klijent

- Povratna vrednost : Indikacija uspešnosti dobavljanja stanja

- `cpe_err_t cpe_set_status_callback(void * const ctx_ptr,
cpe_status_change_callback_t const callback);`
 - Opis: Postavljanje funkcije koja će se pozvati svaki put kada CPE klijent pređe u novo stanje
 - Parametri:
 - ctx_ptr - Kontekst CPE klijenta
 - callback - Funkcija koju treba pozivati
 - Povratna vrednost : Indikacija uspešnosti postavljanja funkcije

4.1.2 libcpe_config_handling.h

- `cpe_err_t cpe_save_config(cpe_ctx_t* ctx_ptr);`
 - Opis: Čuvanje konfiguracije CPE klijenta na trajnu memoriju
 - Parametri: ctx_ptr - Kontekst CPE klijenta
 - Povratna vrednost: Indikacija uspešnosti procesa čuvanja

- `cpe_err_t cpe_load_config(cpe_ctx_t* ctx_ptr);`
 - Opis: Učitavanje konfiguracije CPE klijenta sa trajne memorije
 - Parametri: ctx_ptr - Kontekst CPE klijenta
 - Povratna vrednost: Indikacija uspešnosti procesa učitavanja

- `cpe_err_t cpe_load_conn_data(cpe_conn_data_t *conn_data);`
 - Opis: Učitavanje podataka potrebnih za vezu sa ACS-om
 - Parametri: conn_data - Struktura koja sadrži podatke potrebne za vezu sa ACS-om
 - Povratna vrednost: Indikacija uspešnosti procesa učitavanja

4.2 TR-069 komunikacioni modul

Funkcionalnosti ovog modula omogućavaju uspostavljanje veze sa ACS-om, kao i upravljanje pomoćnim nitima potrebnim za komunikaciju.

4.2.1 libcpe_engine.h

- `void cpe_perform_state_transition(cpe_ctx_t* ctx_ptr,
cpe_evententry_t* event,
cpe_engine_state_t* eng_state) ;`
 - Opis: Promena stanja u mašini stanja za izvođenje TR-069 sesije

- Parametri:
 - ctx_ptr - Kontekst CPE klijenta
 - event - događaj koji se desio u mašini stanja
 - eng_state – trenutno stanje u kom se mašina nalazi

4.2.2 libcpe_comm.h

- `void* cpe_comm_thread(void* data);`
 - Opis: Nit korišćena za prenos HTTP paketa između CPE-a i ACS-a
 - Parametar: data – Kontekst CPE klijenta koji vrši komunikaciju

- `void* cpe_get_cpe_context(void);`
 - Opis: Dobavljanje konteksta CPE klijenta koji koriste drugi delovi TR-069 komunikacionog modula
 - Povratna vrednost: Pokazivač na strukturu konteksta CPE klijenta.

4.2.3 libcpe_int.h

- `void* cpe_engine_thread(void* ctx);`
 - Opis: Nit korišćena za mašinu stanja pri izvođenju TR-069 sesije
 - Parametar: data – Kontekst CPE klijenta koji vrši komunikaciju

- `cpe_err_t cpe_ctx_init(cpe_ctx_t * const ctx_ptr);`
 - Opis – Inicijalizacija pomoćnih struktura potrebnih za rad CPE klijenta
 - Parametar : ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost: Indikacija uspešnosti inicijalizacije

- `cpe_err_t cpe_ctx_cleanup(cpe_ctx_t * const ctx_ptr);`
 - Opis – Deinicijalizacija pomoćnih struktura potrebnih za rad CPE klijenta
 - Parametar : ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost: Indikacija uspešnosti deinicijalizacije

- `cpe_err_t cpe_trigger_post(cpe_ctx_t *ctx_ptr, cpe_trigger_t trigger,`
`void* user_data, char*command_key);`
 - Opis – Objavljivanje događaja koji pokreće TR-069 sesiju
 - Parametri:
 - ctx_ptr – Kontekst CPE klijenta
 - trigger – Tip događaja na krajnjem uređaju

- user_data – Dodatni podaci vezani za događaj
 - command_key – Identifikator događaja
- Povratna vrednost : Indikacija uspešnosti objavljivanja događaja

- `cpe_err_t cpe_insert_after_reboot_trigger(cpe_ctx_t *ctx_ptr,
 cpe_trigger_t trigger, void* user_data,
 char*command_key);`
 - Opis – Objavljivanje događaja koji će se obraditi posle ponovnog startovanja uređaja
 - Parametri:
 - ctx_ptr – Kontekst CPE klijenta
 - trigger – Tip događaja na krajnjem uređaju
 - user_data – Dodatni podaci vezani za događaj
 - command_key – Identifikator događaja
 - Povratna vrednost : Indikacija uspešnosti objavljivanja događaja

- `cpe_err_t cpe_event_post(cpe_ctx_t *ctx_ptr, cpe_event_t event,
 void* user_data);`
 - Opis – Objavljivanje događaja na osnovu kog dolazi do promene stanja u mašini stanja za TR-069 sesiju.
 - Parametri:
 - ctx_ptr - Kontekst CPE klijenta
 - event – Tip događaja koji se desio
 - user_data - Dodatni podaci vezani za događaj
 - Povratna vrednost : Indikacija uspešnosti objavljivanja događaja

- `cpe_err_t cpe_xfer_post(cpe_ctx_t *ctx_ptr, cpe_parser_func
 parser);`
 - Opis – Objavljivanje transfera koji će se sledeći izvršiti.
 - Parametri –
 - ctx_ptr – Kontekst CPE klijenta
 - parser - Funkcija koja će parsirati odgovor na dati transfer.
 - Povratna vrednost : Indikacija uspešnosti objavljivanja transfera

- `cpe_bool_t cpe_xfer_wait(cpe_ctx_t *ctx_ptr, cpe_xferentry_t*
 xfer);`
 - Opis – Čekanje na sledeći transfer koji treba da se izvrši

- Parametri –
 - ctx_ptr – Kontekst CPE klijenta
 - xfer – Transfer koji će se izvršiti kad se objavi.
 - Povratna vrednost – 1 Ukoliko postoji transfer, 0 ukoliko ne postoji
-
- void cpe_post_outgoing_request(cpe_to_acs_soap_req_entry_t *req, cpe_ctx_t *cpe_ctx);
 - Opis : Objavljivanje SOAP zahteva za ACS
 - Parametri:
 - req – SOAP zahtev
 - cpe_ctx – Kontekst CPE klijenta
-
- void cpe_post_after_reboot_request(cpe_to_acs_soap_req_entry_t *req, cpe_ctx_t *cpe_ctx);
 - Opis : Objavljivanje SOAP zahteva za ACS koji će se obraditi posle ponovnog pokretanja uređaja
 - Parametri:
 - req – SOAP zahtev
 - cpe_ctx – Kontekst CPE klijenta
-
- void cpe_clear_session_context(cpe_ctx_t *cpe_ctx);
 - Opis : Oslobađanje struktura korišćenih za poslednju TR-069 sesiju
 - Parametar: cpe_ctx – Kontekst CPE klijenta
-
- void cpe_check_and_upgrade(cpe_ctx_t *ctx_ptr);
 - Opis: Provera i izvršavanje ažuriranja programske podrške uređaja
 - Parametar: cpe_ctx – Kontekst CPE klijenta

4.2.4 libcpe_on_change.h

- repo_err_t repo_set_param_cb_comm(const char* url, const char* value, repo_data_node_type_t type, void *additional_data);
- Opis – Objava promene vrednosti parametara na koju se ACS pretplatio.
- Parametri:
 - url – Putanja do parametra kojem je promenjena vrednost
 - value – Nova vrednost parametra

- type – Tip parametra
- additional_data – Dodatni podaci vezani za parametar
- Povratna vrednost : Indikacija uspešnosti obrade promene vrednosti parametra

4.2.5 libcpe_packers.h

U ovoj datoteci se nalaze funkcionalnosti za kreiranje različitih SOAP struktura. Sve funkcije, prikazane u tabeli 5, imaju isti parametar, kontekst CPE klijenta, i vraćaju indikaciju uspešnosti izvršenja.

Ime funkcije	Opis
cpe_create_inform_handle	Kreiranje <i>Inform</i> SOAP zahteva
cpe_create_empty_handle	Kreiranje praznog HTTP paketa
cpe_create_soap_res_handle	Kreiranje SOAP odgovora
cpe_create_soap_req_handle	Kreiranje SOAP zahteva

Tabela 5. Opisi funkcija za kreiranje HTTP zahteva

4.2.6 libcpe_parsers.h

U ovoj datoteci se nalaze funkcionalnosti za parsiranje različitih SOAP struktura. Sve ove funkcije imaju isti parametre, kontekst CPE klijenta i pokazivač na neimenovanu promenljivu gde smeštaju rezultate parsiranja, i vraćaju indikaciju uspešnosti izvršenja kao događaj koji će se koristiti u mašini stanja TR-069 sesije.

Spisak funkcija se nalazi u tabeli 6:

Ime funkcije	Opis
cpe_inform_parser	Parser za <i>InformResponse</i>
cpe_soap_request_parser	Parser za SOAP zahteve
cpe_soap_response_parser	Parser za SOAP odgovore

Tabela 6. Opisi funkcija za parsiranje SOAP struktura

Pored funkcija za parsiranje SOAP struktura implementirane su i funkcije za prijem HTTP paketa i parsiranje HTTP zaglavљa.

- size_t cpe_collect_data(void *ptr, size_t size, size_t nmemb, void* data);
- Opis: Prijem HTTP tela(eng. HTTP body)
- Parametri:
 - Ptr - Primljeni podaci
 - Size – veličina osnovne jedinice primljenih podataka

- Nmemb – broj primljenih jedinica podataka
- Data – dodatne strukture potrebne za čuvanje potrebnih podataka
- Povratna vrednost : Količina primljenih podataka

```
- size_t cpe_collect_headers( void *ptr, size_t size, size_t nmemb,
void *userdata);
```

- Opis: Prijem HTTP zaglavljia(eng. HTTP header)
- Parametri:
 - Ptr - Primljeni podaci
 - Size – veličina osnovne jedinice primljenih podataka
 - Nmemb – broj primljenih jedinica podataka
 - Data – dodatne strukture potrebne za čuvanje potrebnih podataka
- Povratna vrednost : Količina primljenih podataka

```
- cpe_err_t cpe_get_header_name_value(char** name, char** value,
char* str);
```

- Opis: Parsiranja pojedinačnog HTTP zaglavljia
- Parametri:
 - Name – Parsirano ime HTTP zaglavljia
 - Value – Parsirana vrednost HTTP zaglavljia
 - Str – primljeno HTTP zaglavljje
- Povratna vrednost : Indikacija uspešnosti parsiranja

4.2.7 libcpe_sched_file_transfer.h

- cpe_err_t cpe_sched_file_transfer_init(cpe_ctx_t *ctx_ptr,
download_handler_f download_handler, upload_handler_f
upload_handler);
 - Opis: Inicijalizacija rasporedživača za prenose datoteka
 - Parametri:
 - ctx_ptr – Kontekst CPE klijenta
 - download_handler – Funkcija koja obrađuje preuzimanje datoteke
 - upload_handler – Funkcija koja obrađuje postavljanje datoteke
 - Povratna vrednost : Indikacija uspešnosti inicijalizacije
- cpe_err_t cpe_sched_file_transfer_destroy(cpe_ctx_t *ctx_ptr);
 - Opis: Deinicijalizacija rasporedživača za prenose datoteka
 - Parametar: ctx_ptr – Kontekst CPE klijenta

- Povratna vrednost : Indikacija uspešnosti deinicijalizacije

- `cpe_err_t cpe_sched_file_transfer_start(cpe_ctx_t *ctx_ptr);`
 - Opis: Startovanje niti koja nadgleda rasporedživač za prenose datoteka
 - Parametar: ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost : Indikacija uspešnosti startovanja

- `cpe_err_t cpe_schedule_file_transfer(cpe_ctx_t* ctx_ptr, cpe_file_transfer_type_t file_transfer_type, *file_transfer_req_ptr, unsigned int seconds);`
 - Opis: Raspoređivanje novog prenosa datoteke
 - Parametri:
 - ctx_ptr – Kontekst CPE klijenta
 - file_transfer_type – tip prenosa – preuzimanje ili postavljanje datoteke
 - file_transfer_req_ptr – podaci potrebni za novi prenos podataka
 - seconds – Vreme za koje će prenos biti pokrenut
 - Povratna vrednost : Indikacija uspešnosti objavljivanja

4.2.8 libcpe_sched_trigger.h

- `cpe_err_t cpe_sched_trigger_init(cpe_ctx_t *ctx_ptr);`
 - Opis: Inicijalizacija rasporedživača periodičnog okidača TR-069 sesije
 - Parametri: ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost : Indikacija uspešnosti inicijalizacije

- `cpe_err_t cpe_sched_trigger_destroy(cpe_ctx_t *ctx_ptr);`
 - Opis: Deinicijalizacija rasporedživača periodičnog okidača TR-069 sesije
 - Parametri: ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost : Indikacija uspešnosti deinicijalizacije

- `cpe_err_t cpe_sched_trigger_stop(cpe_ctx_t *ctx_ptr);`
 - Opis: Zaustavljanje rasporedživača periodičnog okidača TR-069 sesije
 - Parametri: ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost : Indikacija uspešnosti zaustavljanja

- `cpe_err_t cpe_sched_trigger_start(cpe_ctx_t *ctx_ptr);`
 - Opis: Pokretanje rasporedživača periodičnog okidača TR-069 sesije
 - Parametri: ctx_ptr – Kontekst CPE klijenta
 - Povratna vrednost : Indikacija uspešnosti pokretanja

4.2.9 libcpe_session_retry.h

- `cpe_err_t cpe_post_session_retry(cpe_ctx_t *ctx_ptr);`
 - o Opis: Objavljivanje da je sesija neuspešno završena
 - o Parametri: ctx_ptr – Kontekst CPE klijenta
 - o Povratna vrednost : Indikacija uspešnosti objavljivanja

- `void* cpe_session_retry_thread(void* arg);`
 - o Opis: Nit za računanje vremena retransmisije TR-069 sesije
 - o Parametri: arg – Kontekst CPE klijenta

4.2.10 libcpe_soap_fault.h

- `cpe_err_t cpe_marshall_soap_fault(xmlNodePtr* soap_enve, cpe_soap_fault_cpe_t* fault_struct, const char* soap_header_id);`
 - o Opis: Kreiranje SOAP izveštaja o grešci
 - o Parametri:
 - soap_enve – Korenki čvor za SOAP izveštaj o grešci
 - fault_struct – Struktura sa podacima o grešci
 - soap_header_id – Identifikator SOAP zaglavlja
 - o Povratna vrednost : Indikacija uspešnosti kreiranja

- `cpe_event_t cpe_handle_soap_fault(xpathContextPtr xpathCtx, unsigned* fault_code);`
 - o Opis: Obrada SOAP izveštaja o grešci
 - o Parametri:
 - xpathCtx – Kontekst SOAP strukture
 - fault_code – Primljeni SOAP kod greške
 - o Povratna vrednost : Indikacija uspešnosti obrade

4.2.11 libcpe_soap_handling.h

Funkcionalnosti sadržane u ovoj datoteci su izdvajanje relevantnih informacija za SOAP zahteva i SOAP odgovora njihova obrada.

Spisak funkcija nalazi se u tabeli 7:

Ime funkcije	Opis funkcije
<code>cpe_get_request_from_soap</code>	Izdvajanje informacija iz SOAP zahteva
<code>cpe_handle_soap_request</code>	Obrada SOAP zahteva

cpe_get_response_from_soap	Izdvajanje informacija iz SOAP odgovora
cpe_handle_soap_response	Obrada SOAP odgovora

Tabela 7. Opisi funkcija za rukovanje SOAP strukturama

4.3 Modul za RPC metode

U okviru ovog modula implementirane su funkcionalnosti potrebne za obradu RPC metoda. Budući da je format funkcija sličan dat je skraćen opis modula. Svaka RPC metoda je obrađena sa tri funkcije:

- cpe_extract_ - Funkcija za izdvajanje podataka iz SOAP strukture
- cpe_handle_ - Funkcija za obradu pozvane RPC metode
- cpe_mars_ - Funkcija za pakovanje odgovarajućeg odgovara na pozvanu RPC metodu

Pritom sve funkcije u modulu imaju iste funkcije imaju iste prefikse, razlikuju se samo u imenu RPC metode na kraju. Primer za RPC metode *DeleteObject*:

- cpe_extract_delete_object
- cpe_handle_delete_object
- cpe_mars_resp_delete_object

4.4 Modul za rukovanje repozitorijumima

Funkcionalnosti ovog modula omogućavaju rukovanje i obradu podataka koji se šalju putem TR-069 protokola, kao i integraciju TR-069 klijenta sa programskom podrškom krajnjeg uređaja.

4.4.1 libcpe_data_repository_engine.h

- repo_err_t data_repo_init(void);
 - Opis: Inicijalizacija modula za rukovanje repozitorijumima.
 - Povratna vrednost: Indikacija uspešnosti inicijalizacije
- repo_err_t data_repo_term(void);
 - Opis: Deinicijalizacija modula za rukovanje repozitorijumima.
 - Povratna vrednost: Indikacija uspešnosti deinicijalizacije
- repo_err_t data_repo_instantiate(unsigned *repo_id, const char* model_filename, const char* initialization_filename);
 - Opis: Instanciranje određenog repozitorijuma na osnovu modela podataka
 - Parametri:

- repo_id – Identifikator repozitorijuma potreban za dalji pristup podacima
- Model_filename – Putanja u sistemu datoteka za XML datoteke koji definiše model podataka
- Initialization_filename – Putanja u sistemu datoteka za XML datoteke koji definiše početne podatke za dati repozitorijum
- Povratna vrednost : Indikacija uspešnosti instanciranja repozitorijuma

- ```
repo_err_t data_repo_register_callback(unsigned repo_id,
 unsigned subscriber_id,
 repo_set_param_cb set_cb, repo_add_object_cb add_cb,
 repo_delete_object_cb delete_cb);
```
- Opis: Prijavljivanje pretplatnika na događaju od određenom repozitorijumu
- Parametri:
  - repo\_id - Identifikator repozitorijuma
  - subscriber\_id - Identifikator pretplatnika
  - set\_cb – Povratna funkcija za događaj postavljanja vrednosti parametra
  - add\_cb – Povratna funkcija za događaj dodavanja podstabla u repozitorijum
  - delete\_db – Povratna funkcija za događaj uklanjanja podstabla iz repozitorijuma
- Povratna vrednost : Indikacija uspešnosti prijavljivanja pretplatnika
  
- ```
repo_err_t data_repo_unregister_callback(unsigned unregister_id);
```
- Opis: Odjavljivanja pretplatnika na događaje u repozitorijumu
- Parametar: unregister_id – Identifikator pretplatnika
- Povratna vrednost – Indikacija uspešnosti odjavljivanja

- ```
repo_err_t data_repo_set_param_value(unsigned setter_id, const
 char* URL,
 const char* value, repo_data_node_type_t data_type,
 char* error_desc);
```
- Opis: Postavljanje vrednosti parametra u repozitorijumu
- Parametri:

- setter\_id – Identifikator entiteta koji postavlja vrednost parametra
  - URL – putanja do parametra u stablu
  - Value – Nova vrednost parametra
  - Data\_type – Tip podatka za novu vrednost parametra
  - Error\_desc – Detaljni opis greške ukoliko dođe do nje u tokom postavljanja vrednosti parametra
  - Indikacija : Indikacija uspešnosti postavljanja vrednosti
- `repo_err_t data_repo_validate_set_param(const char * url, const char *value,`  
`repo_data_node_type_t data_type, char * fault_string);`
- Opis: Provera ispravnosti zahteva za postavljanje vrednosti parametra
  - Parametri:
    - URL – putanja do parametra u stablu
    - Value – Vrednost parametra koja se proverava
    - Data\_type – Tip podatka za novu parametra
    - Error\_desc – Detaljni opis greške ukoliko dođe do nje u tokom provere zahteva
  - Indikacija : Indikacija uspešnosti provere zahteva
- `repo_err_t data_repo_get_param_value(unsigned getter_id, const char* URL,`  
`char** value, repo_data_node_type_t* data_type);`
- Opis: Dobavljanje vrednosti parametra
  - Parametri:
    - getter\_id – Identifikator entiteta koji dobavlja vrednost parametra
    - URL – putanja do parametra u stablu
    - Value – Dobavljena vrednost parametra
    - Data\_type – Tip podatka dobavljene vrednosti parametra
  - Indikacija : Indikacija uspešnosti dobavljanja vrednosti parametra
- `repo_err_t data_repo_get_param_value_list(unsigned getter_id,`  
`const char* URL,`  
`repo_param_value_list_t* param_list, char* fault_string);`
- Opis: Dobavljanje svih vrednosti parametara iz podstabla
  - Parametri :
    - getter\_id – Identifikator entiteta koji dobavlja vrednosti parametara

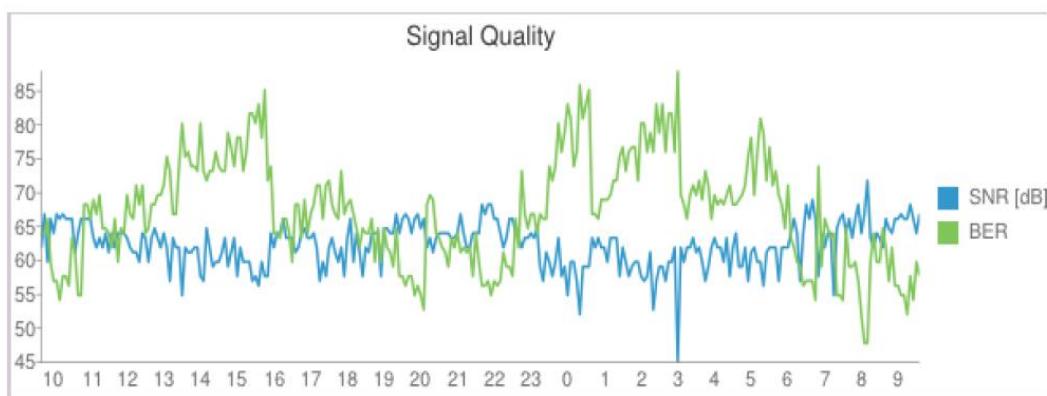
- URL – putanja do čvora u stablu
  - param\_list – Lista dobavljenih vrednosti parametara
  - fault\_string – Opis greške koja se desila prilikom dobavljanja vrednosti
  - Povratna vrednost : Indikacija uspešnosti dobavljanja vrednosti parametara
- 
- repo\_err\_t data\_repo\_set\_param\_attrs(unsigned setter\_id, const char\* URL, cpe\_bool\_t notification\_change, const char\* notification, cpe\_bool\_t access\_list\_change, repo\_access\_list\_t access\_list, char\* fault\_string);
  - Opis : Postavljanje vrednosti atributa za parametre u repozitorijumu
  - Parametri:
    - setter\_id – Identifikator entiteta koji postavlja vrednosti atributa
    - URL – Putanja do čvora u stablu
    - notification\_change – Kontrolna promenljiva koja ukazuje da li treba da se promeni vrednost atributa za notifikaciju.
    - Notification – Nova vrednost atributa za notifikaciju
    - access\_list\_change - Kontrolna promenljiva koja ukazuje da li treba da se promeni vrednost atributa za listu entiteta koji imaju pravo pristupa određenom parametru.
    - access\_list – Lista entiteta koji smeju da pristupe parametru
    - fault\_string – Opis greške ukoliko se ona desi prilikom postavljanja atributa
  - Povratna vrednost : Indikacija uspešnosti postavljanja vrednosti atributa parametara
- 
- repo\_err\_t data\_repo\_get\_param\_attrs\_list(unsigned getter\_id, const char\* URL, repo\_param\_attr\_list\_t\* param\_list, char\* fault\_string);
  - Opis: Dobavljanje vrednosti atributa parametara u stablu
  - Parametri:
    - getter\_id - Identifikator entiteta koji dobavlja vrednosti atributa
    - URL – Putanja do čvora u stablu
    - param\_list – Lista vrednosti svih atributa u podstabalu

- fault\_string – Opis greške ukoliko dođe do nje prilikom dobavljanja vrednosti atributa
  - Povratna vrednost : Indikacija uspešnosti dobavljanja vrednosti atributa
- repo\_err\_t data\_repo\_get\_param\_names\_list(unsigned getter\_id, const char\* URL, cpe\_bool\_t next\_level, repo\_param\_info\_list \*param\_list, char\* fault\_string);
  - Opis: Dobavljanje putanja svih čvorova za određeno postablo
  - Parametri:
    - getter\_id – Identifikator entiteta koji dobavlja putanje čvorova
    - next\_level – Indikator da li se zahtevaju putanje za sve čvorove u podstablu ili samo za one koje se nalaze na prvom nivou ispod čvora zahtevanog promenljivom URL.
    - Param\_list – Lista putanja čvorova u podstablu
    - Fault\_string – Opis greške ukoliko dođe tokom dobavljanja putanja
  - Povratna vrednost : Indikacija uspešnosti dobavljanja putanja čvorova
- repo\_err\_t data\_repo\_get\_inform\_params(unsigned getter\_id, repo\_param\_value\_list\_t\* inform\_list);
  - Opis: Dobavljanje vrednosti parametara obaveznih za *Inform* zahtev
  - Parametri:
    - getter\_id – Identifikator entiteta koji dobavlja vrednosti parametara
    - inform\_list – Lista vrednosti parametara za *Inform* zahtev
  - Povratna vrednost : Indikacija uspešnosti dobavljanja vrednosti parametara
- repo\_err\_t data\_repo\_add\_object(unsigned adder\_id, const char\* URL, unsigned\* added\_node\_id, char\* fault\_string);
  - Opis: Dodavanje podstabla u repozitorijum
  - Parametri:
    - adder\_id – Identifikator entiteta koji dodaje podstablo
    - URL – Putanja do čvora u stablu
    - added\_node\_id – Index novo-dodatog čvora koji je koren podstabla
    - fault\_string – Opis greške ukoliko dođe tokom dodavanja postabla
  - Povratna vrednost : Indikacija uspešnosti dodavanja podstabla

- 
- `repo_err_t data_repo_remove_object(unsigned remover_id, const char* URL,  
 char* fault_string);`
    - Opis: Uklanjanje podstabla iz repozitorijuma
    - Parametri:
      - remover\_id - Identifikator entiteta koji uklanja podstablo
      - URL – Putanja do čvora u stablu
      - fault\_string – Opis greške ukoliko do nje dođe tokom uklanjanja postabla
    - Povratna vrednost : Indikacija uspešnosti uklanjanja podstabla
  
  - `repo_err_t data_repo_set_parameter_key(unsigned setter_id, const char* value,  
 char* fault_string);`
    - Opis: Postavljanja vrednosti identifikatora vrednosti parametara u modulu
    - Parametri:
      - setter\_id – Identifikator entiteta koji postavlja vrednost
      - value – Nova vrednost identifikatora
      - fault\_string – Opis greške ukoliko do nje dođe tokom postavljanja vrednosti identifikatora
    - Povratna vrednost : Indikacija uspešnosti postavljanja vrednosti identifikatora

## 5. Rezultati

TR-069 klijent realizovan u okviru ovog rada je korišćen u okviru razvojnog projekta Insight kao i u okviru studentskih projekata na predmetu Projektovanje namenskih računarskih struktura 1.Jedan od slučajeva upotrebe TR-069 klijenta u okviru Insight razvojnog projekta, koji se odnosi na nadgledanje vrednosti kvaliteta televizijskog signala je prikazan na slici 7.



Slika 7. Prikaz promene vrednosti parametara kvaliteta signala u Insight projektu

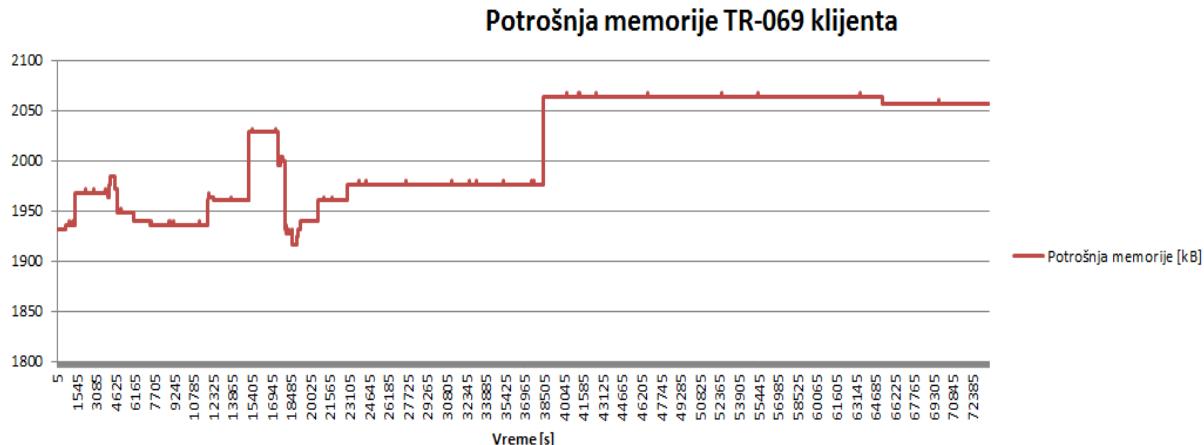
Prilikom razvoja za testiranje modula biblioteke je korišćeno okruženje Cunit, komunikacioni modul je testiran korišćenjem InsightACS 1.0 poslužioca, a za testove opterećenja je korišćena aplikacija za testiranje koja simulira rad STB uređaja.Takođe, tokom razvoja korišćen je razvojni alat Valgrind[6] za detekciju curenja memorije.

### 5.1 Test oprerećenja

Prilikom izvođenja testova operećenja nadgledana je potrošnja operativne memorije i procesorskog vremena od strane TR-069 klijenta.Aplikacija koja testira rad TR-069 klijenta

se zasniva na inicijalizaciji klijenta i kreiranju zasebne niti koja u slučajno odabranim vremenskim trenucima poziva funkcionalnosti klijenta. Testirane funkcionalnosti oslikavaju najčešće slučajeve korišćenja STB:

- Promena kanala
- Promena parametara kvaliteta televizijskog signala
- Dodavanje i uklanjanje kanala iz liste kanala na prijemniku.



Slika 8. Potrošnja memorije TR-069 klijenta tokom izvođenja testova opterećenja

Kao što se može videti sa slike 8. prilikom rada TR-069 nije došlo do curenja memorije. Razlike u potrošnji memorije postoje zbog dodavanja i uklanjanja kanala, što se u radu klijenta oslikava u proširivanju stabla koje predstavlja instancu određenog modela podataka, u ovom slučaju TR-135.

## 5.2 Testiranje modula

Prilikom testiranja modula TR-069 klijenta korišćena je aplikacija zasnovana na CUnit okruženju za testiranje. Izgled aplikacije je prikazan na slici 9.

```

LIBCPE MODULE TESTING

Please choose module which you want to test:

1.TR135
2.TR106
3.DataRepositoryEngine
q/Q-Exit.
```

Slika 9. Izgled aplikacije za testiranje modula TR-069 klijenta

Ulaz u svaki od testova su datoteke:

- Konfiguraciona datoteka biblioteke
- Datoteka modela podataka koji je potrebno instancirati
- Inicijalizaciona datoteke za svakiinstancirani model podataka

Na početku testa se instanciraju repozitorijumi i potom se testiraju funkcionalnosti modula. U slučaju TR106 I TR135 testiranja vrše se pozivi modula za upravljanje repozitorijumima ili prilagodnog sloja uređaja i proveravaju se povratne vrednosti. Prilikom testiranja modula za rukovanje repozitorijumima testiraju se funkcionalnosti za:

- Inicijalizaciju i deinicijalizaciju modula
- Instanciranje i brisanje repozitorijuma
- Registrovanje i odjavljivanje pretplatnika na repozitorijum
- Postavljanje i dobavljanje vrednosti parametara I njihovih atributa
- Dobavljanje strukture podstabala repozitorijuma
- Dodavanje i uklanjanje višestrukog objekta
- Dobavljanje vrednost obaveznih parametara za *Inform* zahtev

Ispravnost izvršenja funkcionalnosti se testira proverom povratne vrednosti i proverom da li se zahtev ostvario promenom u strukturi podstabla. Prilikom završetka testa proveravaju se tvrđenja i ispisuje ukoliko je došlo do greške. Na kraju testiranja ispisuje se rezultat izvršenja svih testova, kao što je prikazano na slici 10.

| --Run Summary: | Type    | Total | Ran  | Passed | Failed |
|----------------|---------|-------|------|--------|--------|
|                | suites  | 1     | 1    | n/a    | 0      |
|                | tests   | 23    | 23   | 23     | 0      |
|                | asserts | 4524  | 4524 | 4524   | 0      |

Slika 10. Rezultati izvršavanja testa za modul za rukovanje repozitorijumima

### 5.3 Testiranje komunikacionog modula

Prilikom razvoja komunikacionog modula korišćen je InsightACS v0.5, InsightACS v1.0 i OpenACS. Ulaz za testiranje je HTTP paket dobijen od strane poslužioca koji sadrži poziv RPC metode na klijentu. Pritom se koriste funkcionalnosti svih modula u TR-069 klijentu. Izlaz iz testa je HTTP paket koji sadži odgovor za datu RPC metodu. Taj HTTP paket se analizira na strani poslužioca i ispisuje se uspešnost izvršenja testa. Prikaz rezultata izvršavanja skupa testova prikazan je na slici 11.

The screenshot shows a web-based application interface for testing. At the top, there's a header with the INSIGHT logo and a 'Version 0.5' link. Below the header, there are two navigation links: 'CPE List' and 'Firmware Upgrade'. The main content area is titled 'Test Cases List' and contains a table with five rows of test cases. The table has columns for ID, TestCaseName, Status, LastTimePassed, LastTimeFailed, LastError, and RunTest. Each row includes a 'Test' button. The test cases are:

| ID | TestCaseName                 | Status          | LastTimePassed         | LastTimeFailed         | LastError | RunTest               |
|----|------------------------------|-----------------|------------------------|------------------------|-----------|-----------------------|
| 1  | Check Inform                 | Green checkmark | 27-06-2013 at 20:10:19 | 26-06-2013 at 15:12:15 |           | <button>Test</button> |
| 2  | Check GetParameterNames      | Red X           | 26-06-2013 at 15:01:28 | 27-06-2013 at 20:10:24 | Unknown   | <button>Test</button> |
| 3  | Check GetParameterAttributes | Green checkmark | 27-06-2013 at 20:10:29 | 15-05-2013 at 11:07:58 |           | <button>Test</button> |
| 4  | Check GetRPCMethods          | Green checkmark | 27-06-2013 at 20:10:34 | 14-06-2013 at 17:28:21 |           | <button>Test</button> |
| 5  | Check SetParameterValues     | Red X           | 21-06-2013 at 14:18:55 | 27-06-2013 at 20:10:39 | Unknown   | <button>Test</button> |

Slika 11. Prikaz rezultata testiranja TR-069 klijenta korišćenjem InsightACS

## 6. Zaključak

Osnovni zadatak rada je bio realizacija TR-069 klijenta u programskom jeziku C i njegova primena na platformi zasnovanoj na Linuks operativnom sistemu uz korišćenje TR-135 modela podataka. Tokom razvoja rešenja dolazilo je definisanja novih zahteva, koji se najviše tiču modularnosti, lakoće održavanja i proširivosti programske podrške. Tokom realizacije ispunjeni su svi zahtevi.

Dalji razvoj može da se odvija u dva smera. U jednom će se razvijati komunikacioni deo radi podržavanja novih profila TR-069 standarda, a u drugom će se pružati podrška za nove modele podataka podržane od strane TR-069 protokola, kao i integracija biblioteke na raznolike krajnje uređaje. Poseban aspekt obuhvata razvoj profila i proširivanju modela podataka za određene uređaje, koji trenutno nisu podržani TR-069 protokolom.

## 7. Literatura

- [1].....TR-069 (CPE Wan Management Protocol), Broadband Forum,  
[http://www.broadband-forum.org/technical/download/TR-069\\_Amendment-1.pdf](http://www.broadband-forum.org/technical/download/TR-069_Amendment-1.pdf)
- [2].....TR-135 (Data Model for a TR-069 Enabled STB), Broadband Forum,  
<http://www.broadband-forum.org/technical/download/TR-135.pdf>
- [3].....TR-106 (Data Model Template for TR-069-Enabled Devices), Broadband Forum, [http://www.broadband-forum.org/technical/download/TR-106\\_Amendment-2.pdf](http://www.broadband-forum.org/technical/download/TR-106_Amendment-2.pdf)
- [4].....OpenACS ,Open source solution for auto-configuration server,  
<http://sourceforge.net/projects/openacs/>
- [5]..... Saša Radovanović, Norbert Nemet, Mića Ćetković, Milan Z. Bjelica and Nikola Teslić, “Cloud-based framework for QoS monitoring and provisioning in consumer devices”, ICCE Berlin,September 2013,accepted.
- [6] ..... Valgrind, instrumentation framework for building dynamic analysis tools,  
<http://valgrind.org/docs/manual/tech-docs.html>