UNIVERZA V LJUBLJANI
FAKULTETA ZA ELEKTROTEHNIKO

MAGISTRSKO DELO

KOMUNIKACIJSKI PROTOKOLI V ELEKTRONSKEM ŠTEVCU ELEKTRIČNE ENERGIJE

Tomaž ŠČUKA, univ.dipl. inž. el.

Mentor
dr. Janko Drnovšek, univ. dipl. inž. el.

LJUBLJANA, 2006

Kazalo

Seznam slik

Seznam tabel

SEZNAM SIMBOLOV IN KRATIC

Simboli

H	Heksadecimalni zapis

Kratice

ACK	Potrditev (Positive Acknowledgement)
ACSE	OSI metoda za vzpostavitev povezave med dvema aplikacijama (Association Control Service Element)
AD	Analogno - Digitalna
AMR	Avtomatsko odčitavanje števcev (Automated Meter Reading)
APDU	Aplikacijska protokolna podatkovna enota (Application Protocol Data Unit)
API	Programski vmesnik (Application Programming Interface)
ASCII	Ameriški podatkovni standard za izmenjavo informacij (American Standard Code for Information Interchange)
ASDU	Aplikacijska storitvena podatkovna enota (Application Service Data Unit)
ASO	Aplikacijski storitveni objekt (Application Service Object)
CF	Kontrolna funkcija (Control Function)
COM	Microsoft-ova platforma za programske komponente (Component Object Model)
COSEM	Dopolnilni standard za merjenje energije (COmpanion Specification for Energy Metering)
CS	Tokovna zanka
CTT	Orodje za preizkušanje skladnosti (Conformance Test Tool)
DCE	Oprema za prenos podatkov (Data Communication Equipment)
DCOM	Nadgradnja COM specifikacije (Distributed Component Object Model)
DLC	Storitev podatkovnega povezovalnega nivoja (Data Link Control)
DLMS	Sistem sporočil za izmenjavo podatkov in kontrolnih informacij med napravami (Device Language Message Specification)
DLMS-UA	Združenje uporabnikov DLMS (DLMS User Association)
DTE	Registrirna naprava (Data Terminal Equipment)
EEPROM	Električno brisljivi programirljivi bralni spomin (Electrically-Erasable Programmable Read-Only Memory)
FRAM	Feromagnetni RAM (Ferroelectric RAM)
FTP	Protokol za prenos datotek (File Transfer Protocol)
GPRS	Storitev za prenos podatkov znotraj GSM omrežja (General Packet Radio Service)
GSM	Globalni sistem za mobilne komunikacije (Global System for Mobile Communications)
GUI	Vmesniško okno (Graphical User Interface)
HDLC	Sinhroni podatkovni povezovalni nivo (High-Level Data Link Control)
HLS	Preverjanje pristnosti visoke varnostne stopnje (High Level Security)
HW	Strojna oprema (Hardware)

IEC	Mednarodna elektrotehniška komisija (International Electrotechnical Commission)
IR	Infra rdeče (Infra Read)
ISDN	Tip telefonskega omrežja namenjenega prenosu zvoka in podatkov (Integrated Services Digital Network)
ISO	Mednarodna organizacija za standardizacijo (International Organization for Standardization)
LAN	Lokalna računalniška mreža (Local area network)
LCD	Zaslon s tekočimi kristali (Liquid Crystal Display)
LDN	Logično ime naprave (Logical Device Name)
LED	Dioda (Light Emitting Diode)
LLC	Logična kontrola povezave (Logical Link Control)
LLS	Preverjanje pristnosti nizke varnostne stopnje (Low Level Security)
LN	Logična imena (Logical Names)
MAC	Kontrola dostopanja medija (Medium Access Control)
M-BUS	Protokol za daljinsko zajemanje podatkov toplotnih števcev (Meter - Bus)
NACK	Zavrnitev (Negative Acknowledgement)
OBIS	Objektni identifikacijski sistem (OBject Identification System)
OLE	Povezovanje in vgrajevanje objektov (Object Linking and Embedding)
OPC	Skupen komunikacijski vmesnik med različnimi napravami pri kontroli in nadzoru tehnoloških procesov (OLE for Process Control)
OS	Operacijski sistem (Operating System)
OSI	Večnivojski model za definiranje delovanja komunikacijskih protokolov (Open Systems Interconnection Reference Model)
PDU	Podatkovna enota protokola (Protocol Data Unit)
PHPDU	Podatkovna enota fizičnega nivoja (Physical Protocol Data Unit)
PICS	Izjava o skladnosti implementacije protokola (Protocol Implementation Conformance Statement)
PIXIT	Dodatne informacije za testiranje implementiranega protokola (Protocol Implementation eXtra Information for Testing)
PPDU	Predstavitvena podatkovna enota protokola (Presentation Protocol Data Unit)
PSTN	Tip telefonskega omrežja (Public switched telephone network)
RAM	Bralno - pisalni spomin (Random Access Memory)
RTOS	Operacijski sistem v realnem času (Real Time OS)
SAP	Dostopna točka servisa (Service Access Point)
SCADA	Sistem za nadzor in zajemanje podatkov (Supervisory Control And Data Acquisition)
SN	Kratka imena (Short Names)
SPDU	Podatkovna enota seje protokola (Session Protocol Data Unit)
SNRM	Tip HDLC podatkovnega okvirja (Set Normal Response Mode)
TCP/IP	Protokola, ki definirata prenos podatkov po omrežjih (TCP - Transmission Control Protocol, IP - Internet Protocol)
TPDU	Transportna podatkovna enota protokola (Transport Protocol Data Unit)
UA	Tip HDLC podatkovnega okvirja (Unnumbered Acknowledge)

UART	Serijski vhodno/izhodni vmesnik (UART - Universal Asynchronous Receiver-Transmitter)
UML	Enotni jezik za modeliranje računalniških sistemov (Unified Modeling Language)
VDEW	Združenje za elektrogospodarstvo (Verband der Elektrizitatswirtschaft)
xDLMS	Razširjeni DLMS (Extended DLMS)
xDLMS-ASE	Servisi razširjenega DLMS aplikacijskega nivoja (Extended DLMS Application Service Element)

1. Povzetek

Merjenje energije pridobiva v zadnjem času vedno večji pomen. Z uvajanjem prostega trga električne energije želi distributer čim boljši pregled nad porabljeno energijo. Zaradi velike prostorske porazdeljenosti takih sistemov je učinkovit način zajemanja energije in upravljanja merilne tehnike ključnega pomena. Zaradi slednje zahteve se je na tem področju pojavila množica komunikacijskih protokolov, ki vsak na svoj način rešujejo omenjeno problematiko. Največji problem, ki se pojavlja pri omenjenih protokolih je ravno univerzalen in poenoten način delovanja. Imeti protokol, ki deluje na napravah različnih proizvajalcev ne glede na posodabljanje naprave je končni cilj vsakega uporabnika komunikacijskih protokolov. Prav tako pomemben člen pri izbiri protokola je kakšen protokol uporabiti. Ali potrebujemo protokol, ki se bo uporabljal v industriji znotraj industrijskih obratov ali pa za gospodinjstvo itd.

V magistrski nalogi bosta podrobneje opisana IEC870-5-102 in DLMS/COSEM komunikacijski protokol. Podrobneje bodo razložene prednosti in slabosti obeh protokolov, gledano s stališča uporabe in implementiranja v elektronski števec električne energije. Sprva bo podana kratka analiza obstoječih komunikacijskih protokolov v elektroindustriji. Sledil bo povzetek o delovanju elektronskih števcev 2. generacije, ki se proizvajajo v Iskraemeco d.d. z namenom lažjega razumevanja integracije omenjenih komunikacijskih protokolov v elektronski števec. Za konec pa bo navedena problematika integriranja komunikacijskih protokolov in dobljeni rezultati.

IEC870-5-102 komunikacijski protokol izhaja iz starejše generacije protokolov in zaradi tega vsebuje številne pomanjkljivosti. Zaradi teh dejstev se umešča v zelo specifične protokole, ki se večinoma uporabljajo za točno določene industrijske aplikacije. Na drugi strani pa predstavlja DLMS/COSEM specifikacija najnovejši protokol, ki opredeljuje univerzalen in široko uporaben komunikacijski kakor tudi funkcionalni protokol (specifikacija določa tudi način delovanja samega števca, ne samo komunikacije). DLMS/COSEM specifikacija predstavlja torej najmodernejši in najuniverzalnejši protokol, ki se danes pojavljajo na trgu električne merilne tehnike.

Abstract

Energy measuring is gaining on its importance recently. With the introduction of opened electrical energy market the demands of electrical distributors are focused on best overview of electricity consumption. Such systems are usually distributed over vast areas, consequently an efficient mean of energy acquisition and equipment control is of great importance. As a result a large group of communication protocols has been developed and each one of them handles the respondent problems accordantly. The biggest problem of all is to produce a universal and uniform communication protocol. To have a protocol which can be used on equipment of different manufacturers regardless of upgrade and maintenance is the main goal of all communication protocol users. An important aspect of choosing the most proper protocol is also the question whether it will be put to domestic or industrial use.

In this master's thesis IEC870-5-102 and DLMS/COSEM communication protocols will be presented, particularly advantages and disadvantages of both protocols looking from code implementation and protocol usage point of view. There is a short analysis of communication protocols in electrical industry at the beginning of the work. Following is the chapter about second generation of electronic meters that are produced in Iskraemeco d.d. This way the integration of communication protocols in electronic meters will be understood better. Problems considering the integration of protocols in to the electronic meter and results can be found at the end of the work.

IEC870-5-102 communication protocol derives from older generations of protocols that is why it contains several faults. Consequently it is labeled as highly specific protocol used only for specific industrial tasks. On the other hand DLMS/COSEM specification represents the newest protocol defining a universal communication functional protocol (specification defines not only the communication sessions but also the way the metering equipment operates). DLMS/COSEM specification is the most modern and universal protocol now days that are available on the market.

2. Uvod

Današnji trg električne energije narekuje čedalje bolj kompleksne predvsem pa prilagodljive števce električne energije, ki so sposobni slediti zahtevam trga. Zanimivo dejstvo je, da si je dandanes v industriji sodobnih števcev električne energije težko predstavljati števec, ki podaja zgolj podatke o porabljeni energiji, pa še ti so dosegljivi le z odčitavanjem stanja pri porabniku. Sodobni elektronski števec električne energije ponuja na tem področju revolucijo.

Sodobni elektronski števec električne energije je kompleksna naprava, sestavljena iz številnih podsestavov. Ti podsestavi zajemajo vhodno/izhodne enote, merilni del, del za obdelavo in shranjevanje merjenih veličin ter njihov prikaz. Na ta način je omogočeno krmiljenje porabe energije, spremljanje številnih dogodkov na števcu in v omrežju, daljinski nadzor nad delovanjem števca in komuniciranje števca z drugimi elektronskimi napravami.

Ravno daljinsko zajemanje podatkov je na področju merjenja porabe električne energije povzročilo razvoj številnih komunikacijskih protokolov. Ti protokoli so nujno potrebni za enoten predvsem pa standardiziran pristop k zajemanju podatkov števca na daljavo.

Skupno vsem komunikacijskim protokolom je, da skušajo na univerzalen način reševati problematiko zajemanja podatkov. Na tem mestu je potrebno omeniti številne protokole, ki se dandanes uporabljajo na področju zajemanja električne energije: IEC62056-21 (nekdanji IEC1107), IEC870-5-102, Euridis (IEC 62056-31) in dragi. Vsi ti protokoli imajo podrobno definiran način prenosa podatkov med napravo in odjemalcem, na žalost pa je interpretacija samih podatkov prepuščena proizvajalcu.

Tudi različne posodobitve protokolov niso med seboj kompatibilne. Na ta način se ruši univerzalnost in medsebojna kompatibilnost sistemov za komuniciranje med različnimi proizvajalci, kar pa izničuje pomen komunikacijskih protokolov. Naprave, ki podpirajo take protokole, v tem primeru elektronski števci električne energije, niso med seboj kompatibilni. Posodabljanje takih sistemov pa je zaradi tega posledično drago in zamudno. Odgovor na nastali problem je bila uvedba novega popolnoma univerzalnega protokola, ki odpravlja tudi slednjo problematiko. Novi protokol je bil razvit pod iniciativo več proizvajalcev števcev električne energije in se imenuje DLMS/COSEM specifikacija.

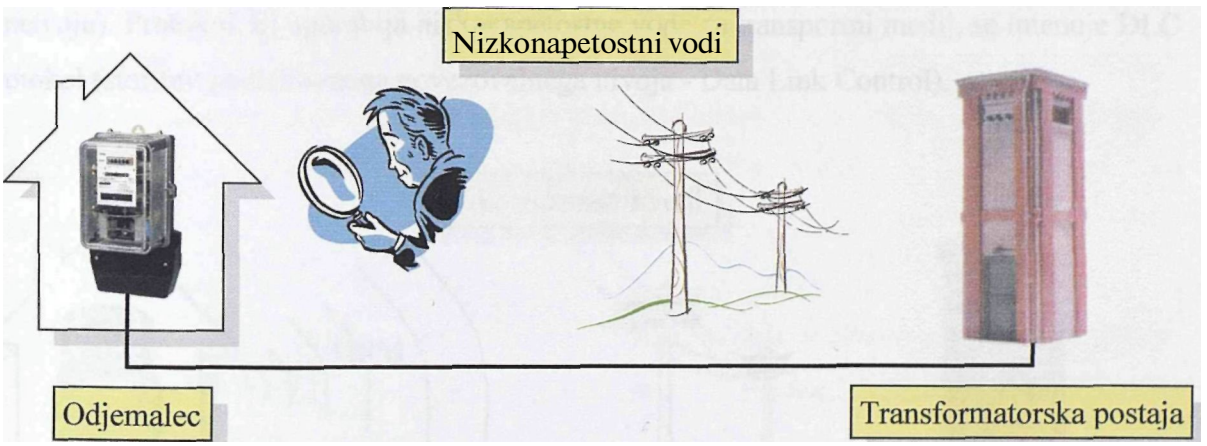
V tem magistrskem delu bosta opisana dva komunikacijska protokola, ki sta dandanes v uporabi. Prvi bo IEC870-5-102 komunikacijski protokol, ki se izrecno uporablja v lokalnih mrežah (LAN) znotraj industrijskih obratov. Ustrezno temu dejstvu je protokol specifičen za SCADA sisteme in je dokaj preprost, brez ustreznih zaščit. Drugi pa je novejši DLMS/COSEM protokol, ki pa ni omejen po namembnosti in se ga lahko uporablja v industriji kot tudi v gospodinjskem sektorju. Predstavljena bo tudi implementacija obeh protokolov na sodobnih elektronskih števcih električne energije. Gre za 2. generacijo števcev električne energije, ki se razvijajo v Iskraemeco *d.d.*

3. Analiza komunikacijskih protokolov uporabljenih v elektronskih števcih električne energije

Zakaj zajemati energijo na daljavo? Avtomatizacija procesov prodira na vsa področja tehnike. Velika prednost, ki jo ponuja avtomatizirano daljinsko zajemanje podatkov o električnih veličinah, ki jih števec lahko meri, se kaže predvsem v prilagodljivosti takega sistema spremembam, posodobitvi, vzdrževanju predvsem pa na sami ceni delovanja.

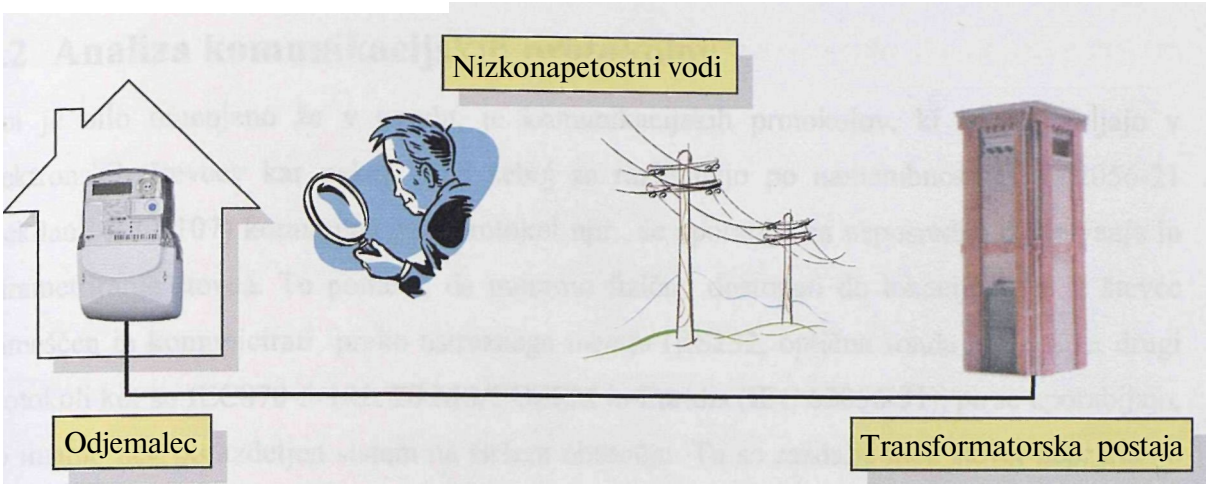
3.1 Razvoj odčitavanja števcev

Glavni in še dandanes pretežno uporabljeni način zajemanja merilnih podatkov števca, predvsem v gospodinjstvih, je ročno odčitavanje. Popisovalec mora fizično dostopati do lokacije kjer se števec nahaja, prebrati vrednost in jo zapisati (slika 1). Takšen način zajemanja podatkov je zamuden, nenatančen predvsem pa drag.



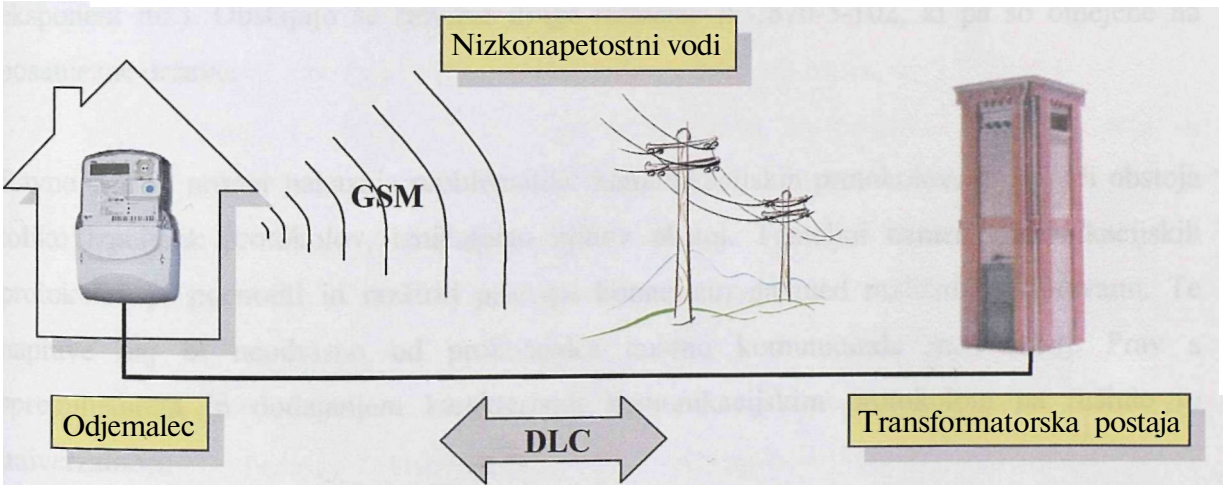
Slika 1. Ročno odčitavanje indukcijskih števcev v preteklosti in danes.

Napredek, ki ga nudi elektronika na tem področju, je ključnega pomena. Kljub prevladujočemu številu indukcijskih števcev, se v gospodinjstvih kot tudi v industriji vedno bolj uveljavlja elektronski števec električne energije (slika 2). Taki števci zasedajo manj prostora so kompaktni, ponujajo možnost obdelave podatkov, hranjenja podatkov, nekateri imajo možnost parametriranja (sprememba merjenih veličin itd.) in navsezadnje imamo lahko tudi možnost ponovnega prepisovanja števčeve aplikacije, v primeru posodobitve programske kode. Vse te prednosti pa pridejo do boljšega izraza, ko lahko daljinsko dostopamo do vseh teh funkcij števca.



Slika 2. Ročno odčitavanje elektronskih števcev.

Daljinsko odčitavanje podatkov je največ prisotno v industriji, v zadnjem času pa se na trgu vedno bolj pojavljajo tudi sistemi za daljinsko odčitavanje gospodinjskih števcev. V tem sektorju se za komunikacijski medij uporablja predvsem nizkonapetostne vode (infrastruktura že obstaja, zato je cenovno najbolj ugodna) in GSM omrežja (protokoli znotraj omrežja so še v razvoju). Protokol, ki uporablja nizkonapetostne vode za transportni medij, se imenuje DLC protokol (storitev podatkovnega povezovalnega nivoja - Data Link Control).



Slika 3. Daljinsko odčitavanje elektronskih števcev preko nizkonapetostnih vodov (DLC protokol) ali preko GSM omrežja.

Kakorkoli, napredek na področju daljinskega zajemanja podatkov v industriji ali gospodinjstvih diktira razvoj in implementacijo številnih komunikacijskih protokolov.

3.2 Analiza komunikacijskih protokolov

Kot je bilo omenjeno že v uvodu, je komunikacijskih protokolov, ki se uporabljajo v elektronskih števcev kar nekaj. Med seboj se razlikujejo po namembnosti. IEC62056-21 (nekdanji IEC1107) komunikacijski protokol npr., se uporablja za neposredno odčitavanje in parametriranje števca. To pomeni, da moramo fizično dostopati do lokacije, kjer je števec nameščen in komunicirati preko ustreznega medija (RS232, optična sonda npr.). Spet dragi protokoli kot so IEC870-5-102, DLMS/COSEM in Euridis (IEC 62056-31), pa se uporabljajo, ko imamo nek porazdeljen sistem na širšem območju. Tu so razdalje med števci neprimerno večje, zato je daljinsko avtomatizirano zajemanje podatkov nujno. Podobnih komunikacijskih protokolov je še veliko več, toda omenjeni protokoli (IEC870-5-102, IEC62056-21, DLMS/COSEM, Euridis (IEC 62056-31)) se dandanes največkrat pojavljajo v industriji elektronskih števcev električne energije.

Pomembno je še dodati, da obstaja veliko izpeljanih protokolov, ki pa ne nudijo večje univerzalnosti. Za primer podajmo IEC870-5-102 protokol. Na tem področju obstaja veliko različic protokola. STOM npr. - gre za okleščeno opcijo IEC870-5-102, ki nudi nekatere izboljšave (možnost zajemanja informacije o merjeni veličini kot so enote, decimalna mesta, eksponent itd.). Obstajajo še številne druge različice IEC870-5-102, ki pa so omejene na posamezne države.

Ravno slednji primer nakazuje problematiko komunikacijskih protokolov, saj zaradi obstoja toliko izpeljank protokolov, izničujemo njihov obstoj. Temeljni namen komunikacijskih protokolov je poenotiti in razširiti principe komuniciranja med različnimi napravami. Te naprave naj bi neodvisno od proizvajalca enotno komunicirale med seboj. Prav s spreminjanjem in dodajanjem karakteristik komunikacijskim protokolom pa rušimo to univerzalnost.

Eden izmed enotnih protokolov, ki odpravlja vse te težave, je k sreči že razvit. Gre za DLMS/COSEM komunikacijski protokol, ki bo razložen podrobneje v nadaljevanju. Vsekakor gre za pomemben korak k poenotenju komunikacijskih protokolov na področju avtomatiziranega zajemanja električne energije. Kljub vsem prednostim enotnih protokolov pa ti terjajo svojo ceno. Taki protokoli so namreč zelo obsežni. To je razumljivo, saj morajo predvideti številne situacije pri prenosu merilnih podatkov. Ravno obsežnost takega

mehanizma pa zahteva tudi zelo kompliciranо implementacijo protokola na elektronskem števcu električne energije.

Uvedba enotnega komunikacijskega protokola pa še ne predstavlja edinega napredka v razvoju avtomatskega daljinskega odčitavanja števcev. Gre za celosten pristop k problematiki, saj komunikacijski protokol predstavlja le način prenašanja podatkov po fizičnem mediju. Govorimo torej o nudenju celostnega sistema za zajemanje in obdelovanja podatkov o merjenih električnih veličinah, kjer enoten in nezastarajoč komunikacijski protokol nudi sistemsko fleksibilnost in nenazadnje tudi cenovno ugodnost (tak sistem ni potrebno posodabljati v primeru dodajanja novih karakteristik obstoječemu protokolu). Najnovejše smernice kažejo, da se bo v prihodnosti kot tudi že danes, trgu ponujal že popolnoma izdelan sistem, ki vsebuje električne števce z vso podporno infrastrukturo (koncentratorji za zajemanje in posredovanje podatkov v centralo; ves programski del za zajemanje in obdelavo podatkov v sami centrali).

V dveh poglavjih bosta podrobneje opisana dva sistema za avtomatizirano zajemanje merilnih podatkov. Oba sistema sta tesno povezana z vsebino magistrske naloge.

3.3 Sodobni AMR sistemi

AMR (Automated Meter Reading) - Avtomatsko odčitavanje števcev

AMR sistemi ponujajo napredne in učinkovite rešitve na področju avtomatizacije in daljinskega zajemanja podatkov električne energije. AMR sisteme najdemo tako v industriji kot v gospodinjstvih. Pomembno pa je tudi omeniti, da so AMR sistemi v postopku uvajanja na trge električne energije in predstavljajo novost na tem področju.

Sistem za avtomatsko odčitavanje gospodinjskih števcev električne energije sestavljajo števci, ki so opremljeni s komunikacijskim modulom za prenos podatkov po nizkonapetostni močnostni mreži, koncentratorji podatkov v transformatorskih postajah z možnostjo komuniciranja s centrom preko GSM/GPRS-omrežja ter programska in strojna oprema v centru za sprejem in obdelavo podatkov. Sistem deluje v realnem času in je poleg osnovne funkcije, to je odčitavanja števcev električne energije, sposoben integrirati tudi druge energente, krmiliti porabo, sporočati izpade in druge dogodke v mreži, preprečevati krajo in tudi odklopiti porabnika.

Inovacija je v tesni povezavi z uvajanjem prostega trga električne energije in plina za gospodinjske porabnike. Sistem za avtomatsko odčitavanje v realnem času spremlja porabo energentov v gospodinjstvih in s povratnimi informacijami tudi vpliva nanjo. Z modifikacijami ga lahko prilagodimo za vse nivoje porabnikov. Mogoče je vključiti porabo tople vode, porabo plina in drugo.

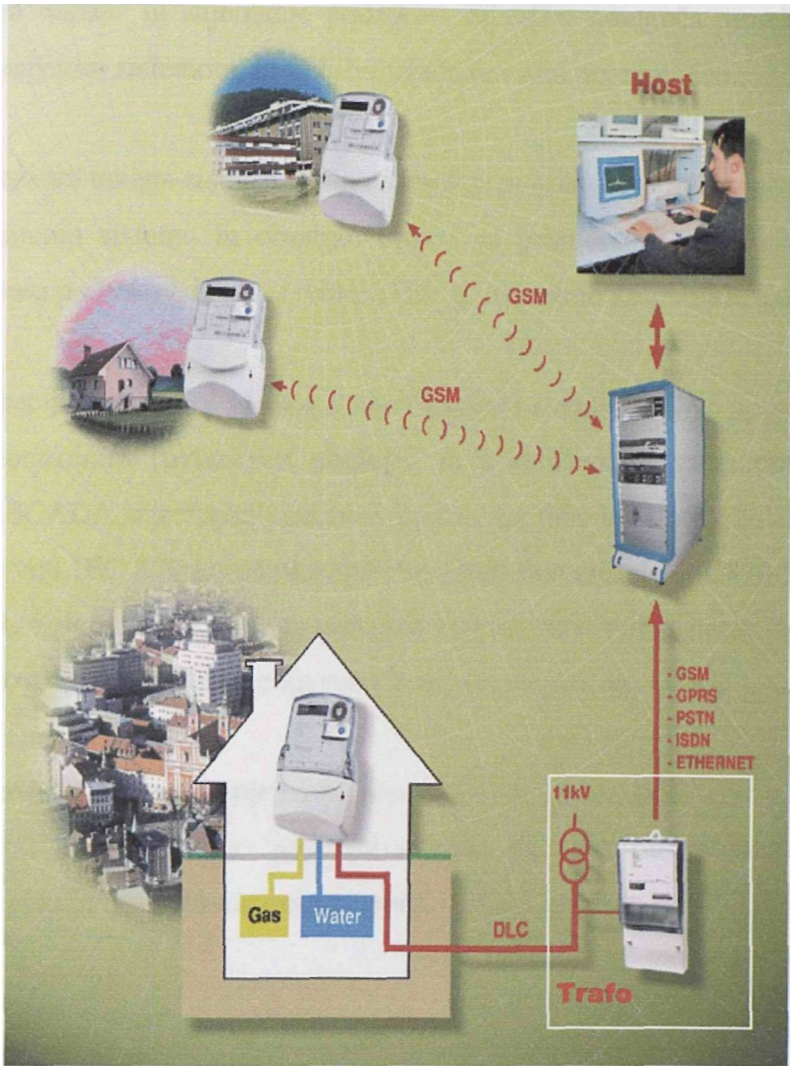
Glavne značilnosti AMR sistemov za gospodinjstvo (podatki temeljijo na ponudbi podjetja Iskraemeco d.d.):

- **Postavitev:** podeželje in urbana področja
- **Števci:** eno in trifazni števci z integriranim komunikacijskim modemom
- **AMR komunikacija:** uporaba DLC in GSM komunikacije
- **Visoko učinkovit DLC:** 99,0 zagotovljeno odčitavanje v manj kot 24h
- **Standardni protokol:** DLMS/COSEM
- **Registracija za obračun** : kumulativa , obremenilna krivulja, dnevno , mesečno
- **Mrežno upravljanje:** avtomatična detekcija števcev
- **Daljinsko programiranje** : parametriranje števec + koncentrator, presnemavanje
- **Več uporabniške funkcije** : vodni in plinski števci (imp. vhodi)

Glavne značilnosti AMR sistemov za industrijo:

- Obračun električne energije in drugih fizikalnih količin (voda, plin, itd)
- Lokalna obdelava in shranjevanje podatkov
- Daljinsko odčitavanje števcev
- Zbiranje podatkov
- Varno shranjevanje merilnih rezultatov v bazo podatkov
- Prenos podatkov v računski center
- Obdelava podatkov
- Generiranje obrazcev za obračun na klasičen in predplačilni sistem

Posredni vplivi inovacije se odražajo v racionalnejši rabi energije, v manjših nihanjih porabe v omrežju, manjših potrebnih proizvodnih kapacitetah energije, v spodbujanju proizvodnje in porabe "zelene" energije. Uvedba AMR-sistemov za vse porabnike energije ima funkcije, ki so izrazito naravnane na racionalno rabo energentov, ki so vključeni v sistem. Vpliv na okolje je torej predvsem v omogočanju zmanjševanja obremenitev okolja.



Slika 4. Sodobni AMR sistem.

Delovanje AMR sistema prikazuje slika 4. Na najnižjem nivoju se nahajajo števci, ki zajemajo podatke o električni energiji ali pa o drugi merjeni veličini. Te podatke se naprej posreduje koncentratorjem (to so naprave, ki zbirajo informacije od števcev), ki se ponavadi nahajajo v transformatorskih postajah. Komunikacija poteka po nizkonapetostnem lokalnem omrežju s pomočjo DLC protokola. Od tu naprej pa koncentratorji pošiljajo podatke v računski center preko GPRS in GSM omrežja.

3.4 SCADA sistemi

V velikih proizvodnih objektih oziroma v obsežnih informacijskih strukturah (povsod kjer je potrebno zajemati veliko količino podatkov) se soočimo s problemom nadziranja takega

sistema. Odgovor na ta problem je SCADA (Supervisory Control And Data Acquisition) oziroma sistem za nadzor in zajemanje podatkov. SCADA omogoča temeljit pregled nad obsežnim informacijskim sistemom, ki bi bil drugače povsem nepregleden.

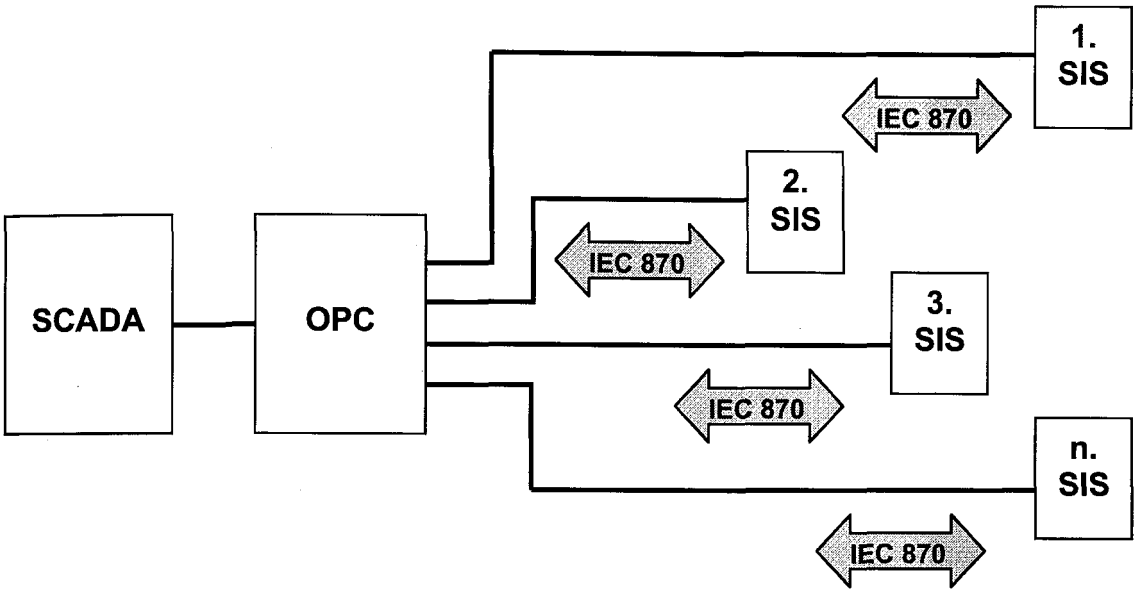
Pomemben del nadzora takega sistema je seveda tudi sam prenos podatkov od nadzorovanega sistema k nadzornemu sistemu in obratno. Pojavi se problem učinkovitega in predvsem poenotenega prenosa podatkov, ki pa je rešen z IEC 870 družino komunikacijskih protokolov.

IEC 870 komunikacijski protokol je tako komunikacijski standard, ki se uporablja za prenos podatkov med kontrolnim **(primarna postaja)** in kontroliranim sistemom **(sekundarna postaja).** Primer SCADA-e je najbolj nazoren prikaz uporabe tega protokola. Pomembno je še poudariti, da izvira IEC 870 komunikacijski protokol (natančneje IEC870-5-102) ravno iz sistemov SCADA. Šele pozneje se je ta protokol razširil na ostale sisteme, kjer je potrebna izmenjava podatkov. Slika 5 prikazuje uporabo protokola v sistemu SCADA.

Pomemben element celotnega omenjenega sistema je tudi prenos podatkov v ustrezni obliki. Sam IEC870-5-102 komunikacijski protokol namreč SCADA ne uporablja, zato se kot vmesnik med števcem in SCADA sistemom uporablja tako imenovani OPC server.

3.4.1 OPC

Za kratico OPC (skupen komunikacijski vmesnik med različnimi napravami pri kontroli in nadzoru tehnoloških procesov) se skriva ime OLE (povezovanje in vgrajevanje objektov - Object Linking and Embedding) for Process Control in predstavlja mednarodni industrijski standard, ki je bil razvit v sodelovanju Microsoft-a in številnih vodilnih proizvajalcev strojne in programske opreme na področju avtomatizacije. OPC temelji na Microsoftovi OLE, COM (Microsoft-ova platforma za programske komponente - Component Object Model), DCOM (Nadgradnja COM specifikacije - Distributed Component Object Model) tehnologiji in predstavlja skupen komunikacijski vmesnik med različnimi napravami pri kontroli in nadzoru tehnoloških procesov.



Slika 5. Uporaba IEC870-5-102 protokola v sistemu SCADA.

V takem sistemu opravlja kontrolirni sistem (SCADA) vlogo izvrševalca in poizvedovalca. Samo na njegovo zahtevo oziroma prošnjo mu kontrolirani sistem (na Sliki 5 so to L, 2., 3. in n. sistem) odgovarja in obvešča. Vsa podatkovna izmenjava se odvija na zahtevo/prošnjo kontrolirnega sistema. Posrednik med dejansko kontrolirano napravo in SCADA sistemom, pa je OPC strežnik. Naloga OPC strežnika je zajemanje določenega nabora parametrov, ki jih SCADA zajema na zahtevo ali ciklično.

4. Elektronski števec električne energije

Vsekakor spada digitalni način merjenja električne energije med moderne metode merjenj te veličine. Prednosti pred analognim zajemanjem električne energije sta predvsem:

- enostavnejša avtomatizacija merjenj
- enostavna obdelava in podajanje merilnih rezultatov

V grobem bi lahko razdelili funkcionalni del elektronskega števca na dva pomembnejša dela. Prvi skrbi za transformacijo in pretvorbo merjene veličine v digitalno, drugi sklop pa poskrbi za digitalno obdelavo merjene veličine. Slednji je implementiran na mikroprocesorskem sistemu, ki dejansko predstavlja sistem za interpretiranje zajetih podatkov, njihovo hranjenje, prikazovanje, včasih pa tudi za pošiljanje preko komunikacijskih poti.
Transformacija merilne veličine poteka preko magnetnega sklopa (analogni del), ki transformira merjeno veličino v obliko primerno za analogno/digitalno pretvorbo. AD (Analogno Digitalna) pretvorba je izvedena na čipu (integrirano vezje), ki poskrbi za celotno pretvorbo analogne informacije v digitalno in na ločenih izhodih poskrbi za zajemanje različnih podatkov o energiji, (delovna, jalova, navidezna), podatki o napetostih in tokovih, faznemu kotu itd. Seveda je nujen del takega vezja tudi analogni del za zajemanje podatkov. V preteklosti in tudi v sedanjem času se še vedno v veliki večini uporabljajo senzorji, ki delujejo na Hallovem principu zajemanja podatkov o električni energiji. Ta način je v sedanjem času že rahlo zastarel saj nam sodobna tehnologija ponuja senzorje, ki so bolj precizni in tudi izhodna veličina je ustrezno prilagojena (linearnost, kompenzacija različnih motilnih vplivov itd.). Najboljši primer takega senzorja je tuljavica Rogowski. Ravno ta tuljavica je bila uporabljena pri gradnji in razvoju števcev 2. generacije (najnovejša družina števcev, ki vključuje nove principe zajemanja podatkov o energiji in modularni princip izgradnje programske in strojne opreme) v Iskraemeco.
Kot je bilo omenjeno že v uvodu, vsebuje elektronski števec veliko podsestavov. Katere podsestave vsebuje, pa zavisi predvsem od tipa števca (industrijski, gospodinjski) pa tudi od kupca števca. Podrobnejši opis strojne in programske opreme sledi v podpoglavjih.

Po funkcionalnosti je elektronski števec veliko več kot le zgolj naprava za merjenje energije. Zahteve na trgu in predvsem razvoj tehnologij (komunikacija in sodobni komunikacijski

protokoli: DLMS, TCP/IP, IEC870, IEC62056-21 itd., vedno bolj zmogljivi mikrokrmilniki in še bi lahko naštevali) prispevajo k visoki stopnji kompleksnosti elektronskih števcev.

Primer: sodobni industrijski števec električne energije mora imeti sposobnost zajemanja podatkov o stanju energije po želenih tarifah (v industriji je ta zajem še bolj kompleksen kot pri gospodinjstvih saj obstaja več tarifnih shem), spremljanje energij po vseh 4 kvadrantih (realna, jalova in navidezna energija), možnost obračuna, ki ga števec izvaja na določena časovna razdobja ali pa se ga lahko tudi aktivira ročno, shranjevanje podatkov o kumulativnih/trenutnih energijah za določeno časovno obdobje, shranjevanje določenih (v naprej domenjenih) dogodkov v števcu, isto lahko tudi velja za napetostni profil itd, števec mora biti parametribilen na samem terenu (če se pogoji ali pa zahteve o merjenju tipa energije spremenijo ni potrebno menjati števcev in kupovati novih), števec mora skrbeti za kredibilnost shranjenih podatkov (različno preverjanje kontrolnih vsot, CRC itd.), sposoben mora biti različnih vrst komunikacij (TCP/IP, IEC870-5-102, IEC62056-21, FTP, DLMS/COSEM itd.), vsebuje tudi strojne vmesnike (module) za komunikacijo (GSM, PSTN modemi, RS232, RS485, CV itd.), preko te komunikacije mora pošiljati različne podatke o stanjih energije, obračunih, znati aktivirati določene zahteve distributerja itd.

4.1 Vrste elektronskih števcev električne energije

Od tega mesta naprej bo opisana družina števcev in vsa podporna oprema (programska kot tudi strojna), ki je tesno povezana z razvojem merilne opreme v podjetju Iskraemeco.

4.1.1 Gospodinjstvo

Elektromehanski in elektronski števci tipično razreda točnosti 2, so namenjeni za merjenje električne energije v gospodinjstvih. Zahtevnejši modeli števcev imajo že vgrajene funkcijske module (moduli za daljinsko komuniciranje z operaterjem: odjem podatkov, parametriranje itd.) za vključevanje v sisteme za odčitavanje in krmiljenje v različnih tarifnih sistemih. Taki števci so praviloma preprostejši in ne omogočajo merjenja energije po vseh kvadrantih. Možne izvedbe so enofazni kot tudi trifazni števci.

Slika 6. Sodobni gospodinjski elektronski števec električne energije.

4.1.2 Industrija in elektrogospodarstvo

Števci električne energije za industrijo in elektrogospodarstvo ustrezajo različnim zahtevam glede merilne točnosti. Omogočajo merjenje delovne in jalove energije ter moči v štirih kvadrantih, registracijo obremenitvenih krivulj ter komuniciranje in prenos podatkov na daljavo. Elektronski števec kot osnovni gradnik sistema merjenja in upravljanja električne energije, se lahko vključi tudi v sistem za vodenje in obračun. Taki sistemi (lahko tudi po nekaj tisoč in več) vsebujejo tudi številne naprave za zajemanje podatkov s števcev, ki jih posredujejo naprej operaterju. Števci električne energije za industrijo in elektrogospodarstvo imajo tipično vrednost merilne točnosti 1, 0.5, 0.2.

Slika 7. Industrijski elektronski števec električne energije.

Nekaj funkcionalnih zmožnosti števca namenjenega industriji in elektrogospodarstvu:

- **Registriranje energije** (Energy and demand registration)
- **Shranjevalnik bremenske krivulje** (Load profile recorder)
  Bremenska krivulja je niz zajetih podatkov (npr. energija) na določeno časovno periodo (1/2, 1 uro itd.)
- **Shranjevalnik dogodkov** (Log book recorder)
  Pomembni dogodki, ki so se zgodili v števcu (obračuni, poskus vdora itd.)
- **Registriranje napetosti in tokov** (Voltage and current registration)
- **Časovna uporaba: tarifne sheme, prazniki** (Time Of Use: tariff program, holidays)
  Delovanje števca (npr. obračun energije) zavisi od tarifnih shem, praznikov
- **Spreminjanje parametrov** (Formatting parameters, Sequencing parameters)
- IEC62056-21, IEC870-5-102, DLMS/COSEM komunikacijski protokol (IEC62056-21, IEC870-5-102, DLMS/COSEM communication protocol)
- **LCD gonilnik po želji kupca** (Custom LCD driver)

4.2 Modularni principi uporabljeni pri gradnji števcev v Iskraemeco

Zaradi boljšega razumevanja problematike vgradnje komunikacijskih protokolov v števce, sledi še kratek opis delovanja elektronskega števca električne energije 2. generacije, podjetja Iskraemeco.

4.2.1 Modularni principi uporabljeni pri strojni opremi

Strojni del je v grobem sestavljen iz:

- magnetnega sklopa
- merilnega dela
- procesorskega dela
- dodatnih modulov (predvsem za komunikacijo)
- periferije (LCD, tipke, IR komunikacija, priključnica)

Magnetni sklop

Tuje mišljen del za zajemanje podatka o toku in napetosti (tuljavica Rogowski, Hallova sonda).

Merilni del

Merilni del je implementiran na merilnem čipu. Na vhod čipa se pripelje signal iz tuljavice, na samem izhodu čipa pa se že nahajajo inkrementi, ki nosijo informacijo o različnih energijah, napetostih, frekvenci itd.

Procesorski del

Procesor oziroma mikrokrmilnik skrbi za sinhronizacijo vseh elementov naprave in upravlja delovanje števca. Na števec so vezani tudi ostali spominski elementi FLASH (spominski medij za večje količine podatkov; počasen dostop), EEPROM, FRAM itd.

Dodatni moduli (predvsem za komunikacijo)

Moduli so grajeni tako, da se jih zgolj namesti na priklopno mesto ("Plug and Play"). So fizični vmesniki za vprogramirane komunikacijske protokole (ti se že nahajajo v števcu).

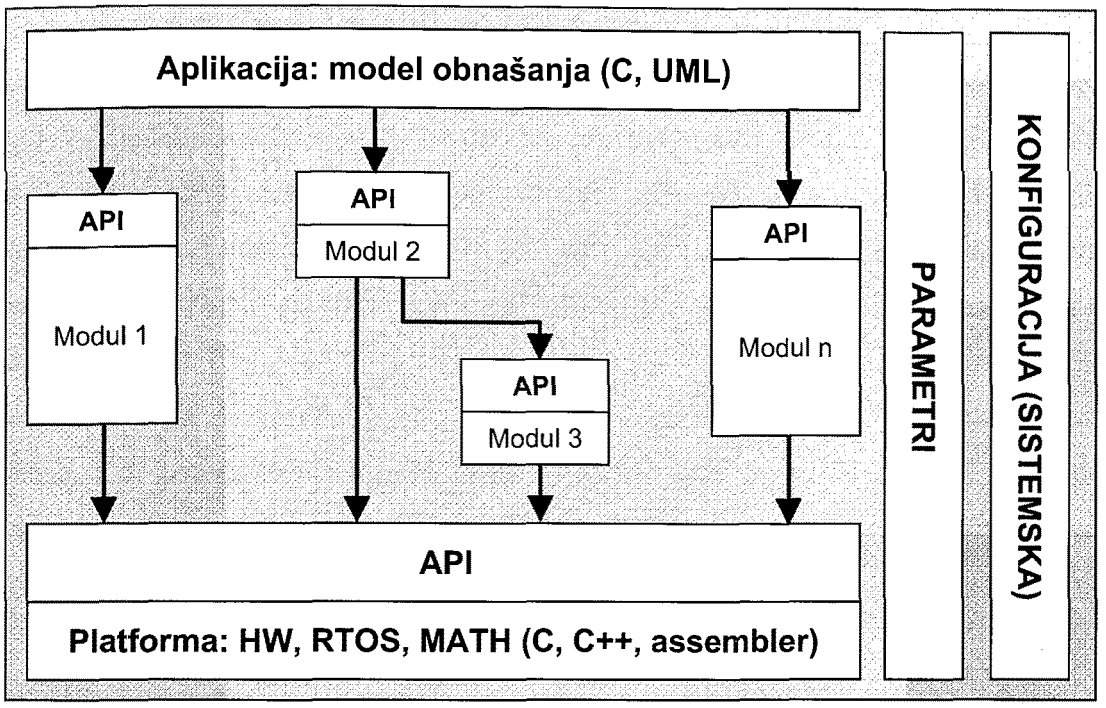
Periferija (LCD, tipke, IR komunikacija, priključnica)

Za prikaz nekaterih (po izbiri) merjenih veličin števca se uporablja LCD zaslon. Za posebne indikacije se uporabljajo tudi LED diode, IR dioda pa se uporablja za optično branje podatkov iz števca, pa tudi za vpisovanje podatkov v sam števec.

4.2.2 Modularni principi uporabljeni v programski kodi

Programski del elektronskega števca je prav tako grajen modularno. Glavni razlogi za uvedbo modularne programske kode v števcih so:

- reciklaža programske kode (isti programski moduli se lahko uporabijo v več števcih)
- enostavnejša delitev razvojnega dela na več izvajalcev
- zmanjševanje rizikov (morebitni zapleti, ki bi ogrozili razvoj števca) pri razvoju novih izdelkov
- skrajševanje razvojnega cikla
- možna razmeroma enostavna menjava platforme (direktni vpliv na ceno izdelka)
- možna simulacija na navidezni platformi (večja zanesljivost programske kode)



K

Slika 8. Modularna zgradba programske kode.

Glede na funkcionalnost, delimo programsko kodo števca na več delov:

Operacijski sistem (OS)

OS oziroma RTOS (OS v realnem času - Real Time OS) uporabljen v števcih MT830 in MT831 je narejen v Iskraemeco (JOS operacijski sistem: J-OS), medtem, ko je operacijski sistem v MT860 ECOS. OS skrbi za pravilno delovanje vseh programskih modulov, platforme, aplikacije, skratka vseh delov programske kode, ki jih zahteva števec za pravilno delovanje.

Platforma

Glavna naloga platforme je, da izolira strojno opremo, operacijski sistem in določene dele programske kode, od splošnih modulov, ki se pojavljajo v različnih napravah (števcih). Platformo sestavlja več delov. Vsak od teh delov omogoča modulom dostop do določenih funkcij, ki jih nudi strojna oprema. Za vsak del platforme je določen programski vmesnik oziroma API (API - Application Programming Interface). Moduli in aplikacija ne smejo dostopati do strojne opreme drugače kot preko tega API-ja.

Moduli

Moduli so gradniki, ki omogočajo določeno funkcionalnost. Funkcija modula je izbrana tako, da tvori zaključeno celoto. Moduli morajo biti praviloma med seboj neodvisni, to pomeni, da po možnosti ne uporabljajo funkcij dragih modulov. Praviloma imajo moduli na razpolago le funkcije, ki jih nudi platforma in tiste ki jih realizirajo sami. Tako na primer ni dopustno, da modul za komunikacijo zahteva za svoje delovanje prisotnost modula za bremensko krivuljo (load profile).

Seveda so izjeme k pravilu: kadar so moduli vertikalno soodvisni je taka soodvisnost dopustna. Tak primer so posamezni nivoji komunikacijskega protokola. IEC 870-5 definira povezovalni nivo za celo vrsto protokolov in je zato lahko podrejen več višjenivojskim protokolom (IEC 870-5-102, M-BUS ...).

Aplikacija

Pri razvoju novih izdelkov se za modeliranje aplikativnega dela programske kode uporablja enotni jezik za modeliranje računalniških sistemov oziroma UML (Unified Modeling Language), ki ga vsebuje programsko orodje Rhapsody. Na osnovi tega modela se generira programska koda aplikacije.

Za tako aplikacijo je potreben tudi operacijski sistem (RTOS). Seveda je možno aplikacijo narediti tudi brez Rhapsody in RTOS. V takem primeru gre običajno za neskončno zanko, ki krožno kliče vse module po vrsti. Problem pri takšni zasnovi je, da se mora klic modula izvršiti do konca, da pride na vrsto naslednji (to vzame velikokrat dosti časa). Moduli morajo zato klice izvajati hitro in daljše procedure interno razbiti na več krajših korakov.

Sistem

Sistemske definicije (konstante), pravila in rešitve (različne konfiguracije funkcionalnosti števca), ki se jih mora zagotoviti za pravilno delovanje sistema (števca).

Parametri

Parametri vseh modulov, platforme in aplikacije, morajo biti združeni v skupno strukturo parametrov sistema (števca). Ti parametri se lahko spreminjajo v delovnem času števca in določajo njegovo delovanje.

5. IEC870 komunikacijski protokol

5.1 Splošni pregled

IEC870 komunikacijski protokol je namenjen zanesljivemu prenosu podatkov za daljinsko kontrolo sistemov. Ni namenjen masovnemu prenašanju podatkov (tako kot recimo TCP/IP protokol), temveč zanesljivemu prenašanju podatkov znotraj industrijskih obratov. Zaradi tega se protokol uporablja predvsem v lokalnih mrežah v industrijskih obratih. Taka vrsta povezave omogoča nižje hitrosti prenosa podatkov od 1200, 9600 itd. do 115.200kBits (skrajna zgornja meja, ki se ponavadi ne uporablja). Gre tudi za protokol, ki je v industriji prisoten že nekaj časa, gre torej za starejši protokol.

Omenjeni protokol sestoji iz več dodatnih protokolov. Naj jih nekaj omenimo:
IEC870-5-101, IEC870-5-102, IEC870-5-103 itd.

Podrobneje bo opisan IEC870-5-102 protokol in njegov fizični nivo (o tem več kasneje) IEC870-5 komunikacijski protokol.

5.2 Zgodovina nastajanja IEC870 protokola

V letu 1980 je bila ustanovljena skupina katere zadolžitev je bila zasnova mednarodnega komunikacijskega protokola namenjenega daljinski kontroli močnostnih električnih prenosnih sistemov (telecontrol of electric power transmission systems).

Sprva so potekale priprave na asinhronskem podatkovnem nosilcu (podatkovnem okvirju), ki bi bil osnova prenosu podatkov. Podatki naj bi se prenašali med kontrolno in kontrolirano postajo (lahko tudi več postaj), ki bi bile med seboj trajno povezane. Sama izmenjava podatkov pa naj bi potekala po sistemu »zahteva - odgovor« (Request-Respond) in »pošlji - potrdi« (Send-Confirm). Vsa ta sporočila pa naj bi vsebovala tudi podatke o naslovih vključenih sistemov (naslov števca itd.).

Zaradi konstantnega napredka in razvoja na področju tehnologij upravljanja na daljavo, je bilo za razvoj standarda porabljenega več časa kot sprva pričakovano. Ni imelo smisla standardizirati tehnologije in protokole, ki so hitro zastareli. Bile pa so tudi zahteve naj se uporabi še ostale standarde s področja komunikacij. Npr.: Uporabljen je bil OSI (večnivojski

model za definiranje delovanja komunikacijskih protokolov - Open Systems Interconnection Reference Model) referenčni model (takrat še v razvoju), ki definira notranjo povezanost samega sistema (na kateremkoli področju: programskem, fizičnem itd.). Notranjo povezanost definira s pomočjo sedmih nivojev (layer) izmed katerih so jih v novem standardu uporabili samo tri (aplikacijski nivo - application layer, povezovalni nivo - link layer, fizični nivo - physical layer).

Po definiranju zahtev za fizični (physical layer) in povezovalni (link layer) nivo, se je začelo delo na aplikacijskim nivoju.

Sprva se je definirala osnova za obliko formata aplikacijskega sporočila (ASDU - Application Service Data Unit). Format je vseboval:

- **glavo** (header): definira tip povezave
- **vzrok pošiljanja** (cause of transmission): definira vzrok povezave
- **natančno določena podatkovna struktura** (detailed structure of the message): vsebuje informacijske objekte (skupek podatkov), ki vsebujejo natančneje informacije (informacijski elementi) o vrsti podatkov, časovnih značkah itd.

Naslednji korak je bila definicija informacijskih elementov, ki je bila tudi izpeljana. Protokol se je sprva imenoval IEC 870-5, ki se je pozneje preimenoval v IEC 60870-5 (v času, ko sta se ISO in IEC številčna sistema uskladila). ISO - Mednarodna organizacija za standardizacijo (International Organization for Standardization).

Pet osnovnih sestavnih delov standarda je bilo javno objavljenih leta 1990. Zaradi dejstva, da se tehnika razvija, je bilo dodano k temu standardu še peščica dopolnilnih, ki uravnavajo vrzel med obstoječimi in novimi tehnologijami. Delo se je nadaljevalo na dopolnilnem standardu IEC60870-5-101 (Basic Telecontrol Tasks - osnovne naloge daljinskega upravljanja), ki je bil uradno predstavljen leta 1995. Hkrati pa sta se razvijala še dva dopolnilna standarda:

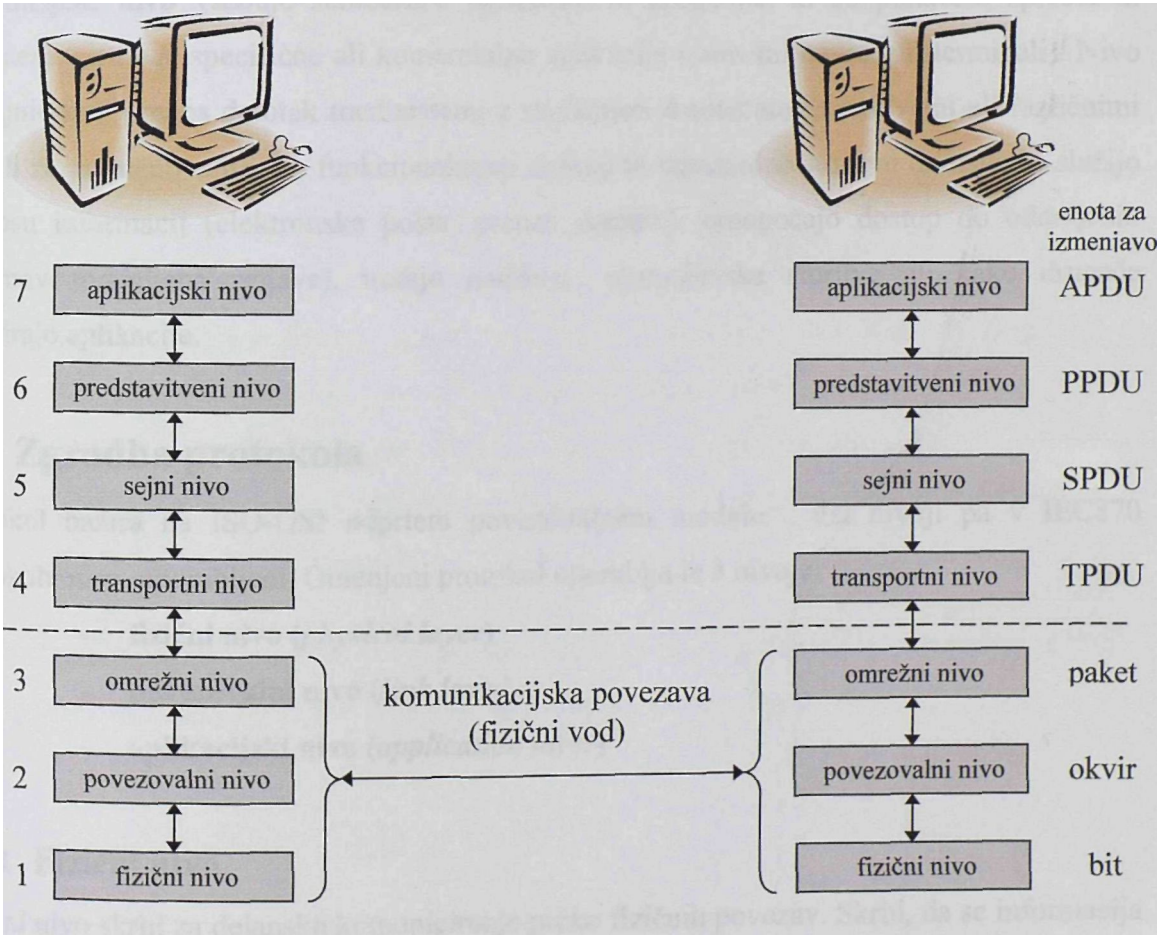
- IEC 60870-5-102: protokol, ki definira prenašanje podatkov med električnim porabnikom in distributerjem, ki uporabljata tak način prenašanja podatkov, kot je definiran v osnovni različici IEC870 protokola

- IEC 60870-5-103: protokol, ki definira informativni vmesnik za prikaz informacij znotraj sistema

Sledilo je nekaj izboljšav in pojasnil protokolu, ki so podrobneje definirali časovne značke. Leta 1996 se je pojavila potreba po dopolnilnem protokolu, ki bi definiral pošiljanje ASDU enot preko interneta. Dopolnilni protokol 60870-5-104 je poskrbel za standard na tem področju in je bil izdan šele pred kratkim.

5.3 OSI referenčni model

OSI referenčni model predstavlja abstraktni opis komunikacijskih nivojev. Funkcionalnost modela je razdeljena na sedem nivojev. Posamezen nivo nudi storitve nivoju nad sabo in uporablja funkcije, ki jih ponuja spodnji. OSI model določa, kako se informacija iz aplikacije na enem računalniku preko omrežja prenese v aplikacijo na drugem računalniku. Strukturo OSI modela prikazuje slika 9.



Slika 9. OSI referenčni model.

Fizični **nivo** je namenjen dejanskemu prenosu bitov informacije po fizičnem mediju (RS232, RS485, CS itd.).

Povezovalni nivo prenaša podatkovne okvirje med dvema uporabnikoma. Nivo pravzaprav ne zanima kakšne podatke prenaša, skrbi le za njihovo zanesljivo dostavo/prejem.

Omrežni nivo skrbi za usmerjanje podatkovnih paketov skozi omrežje. Preprečuje in odpravlja prenasičenost omrežja s paketi.

Transportni nivo. Uporabniške informacijske enote prilagodi v ustrezno obliko primerno transportnemu sistemu. Skrbi tudi za preprečevanje napak pri prenosih (kontrola pretoka - flow control), nasičevanja ponora s podatki, izgube podatkov itd.

Sejni nivoje namenjen storitvam, ki podpirajo logično povezovanje oddaljenih procesov med seboj oziroma vzpostavitev seje med njimi. Skrbi za komunikacijo in sinhronizacijo med procesi,

Predstavitveni nivo se ukvarja predvsem s sintakso in semantiko informacij, ki se prenašajo (ASCII, Unicode kodiranje podatkov). Skrbi torej za združljivost podatkov, ki se prenašajo med različnimi računalniškimi okolji, pa tudi za njihovo zaščito.

Aplikacijski nivo vsebuje standardne aplikacije in protokole, ki so ponavadi splošni in vključeni v mnoge specifične ali komercialne aplikacije (omrežni navidezni terminali). Nivo usklajuje tudi prenos datotek med sistemi z različnimi datotečnimi ureditvami ali različnimi pravili za poimenovanje. Po funkcionalnosti delimo te standardne storitve na tiste, ki služijo prenosu informacij (elektronska pošta, prenos datotek), omogočajo dostop do oddaljenih sistemov (oddaljena prijava), nudijo nadzorno upravljavske storitve ali kako drugače podpirajo aplikacije.

5.4 Zgradba protokola

Protokol bazira na ISO-OSI odprtem povezovalnem modelu. Vsi nivoji pa v IEC870 protokolu niso uporabljeni. Omenjeni protokol uporablja le 3 nivoje:

- **fizični nivo** *(physical **layer)***
- **povezovalni nivo** ***(link layer)***
- **aplikacijski nivo** ***(application layer)***

5.4.1 Fizični nivo

Fizični nivo skrbi za dejansko komuniciranje preko fizičnih povezav. Skrbi, da se informacija (ta del protokola vidi dejansko informacijo kot niz "ničel in "enic") ustrezno sestavi in

odpošlje. Seveda skrbi fizični nivo tudi za sprejemanje podatkov. Vse te podatke pa dobiva oziroma posreduje povezovalnemu nivoju. Omeniti je še potrebno, da temu nivoju ustreza IEC870-5 protokol, katerega naloga je ravno funkcija fizičnega nivoja OSI modela.

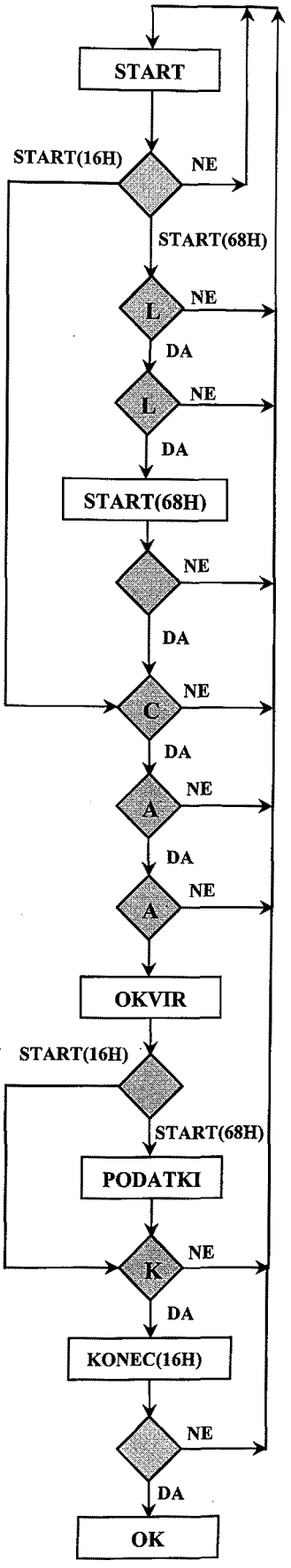
5.4.1.1 Sprejemna procedura

Slika 10 prikazuje delovanje IEC870-5 protokola, ko sprejema podatkovne okvirje iz komunikacijskih vodnikov. Iz bločne sheme je razvidno, da se koda izvaja postopoma. Začne se s sprejemom startnega zloga, ki je lahko začetek spremenljivega (START-68H) ali pa začetek nespremenljivega okvirja (START-10H). V primeru, da gre za začetek spremenljivega okvirja, sledita startnemu zlogu zloga dolžine podatkovnega dela okvirja. V primeru, da dolžina ni veljavna se vrne algoritem na začetek.

Procedura znova pričakuje startni zlog in naprej kontrolni zlog. V tem zlogu se tudi nadaljuje zaporedje nespremenljivega okvirja tako, da se nadaljevanje ujema za oba tipa okvirjev. Kontrolnemu zlogu sledita adresna zloga oziroma adresni zlog, če procesor podpira le enozložne adrese. S pomočjo adresnega zloga algoritem ugotovi ali je namenjen okvir njemu ali ne. V primeru, da algoritem ugotovi, da je okvir namenjen njemu, se začne za spremenljiv okvir prenos podatkov, za nespremenljiv okvir pa pomeni, da se nadaljuje s kontrolno vsoto (K).

Spremenljiv okvir ima toliko podatkov, kolikor je določeno z začetno dolžino L (tu je pomembno omeniti, da je dolžina L sestavljena iz kontrolnega in obeh adresnih zlogov ter iz podatkov) in le toliko podatkov bo sprejel. Ker bo nadaljeval s kontrolno vsoto (K), bo algoritem takoj ugotovil ali se je dogodila napaka pri prenosu in ponovil sprejemno proceduro. V primeru nespremenljivega okvirja pa je kontrolna vsota sestavljena le iz kontrolnega in vseh adresnih zlogov.

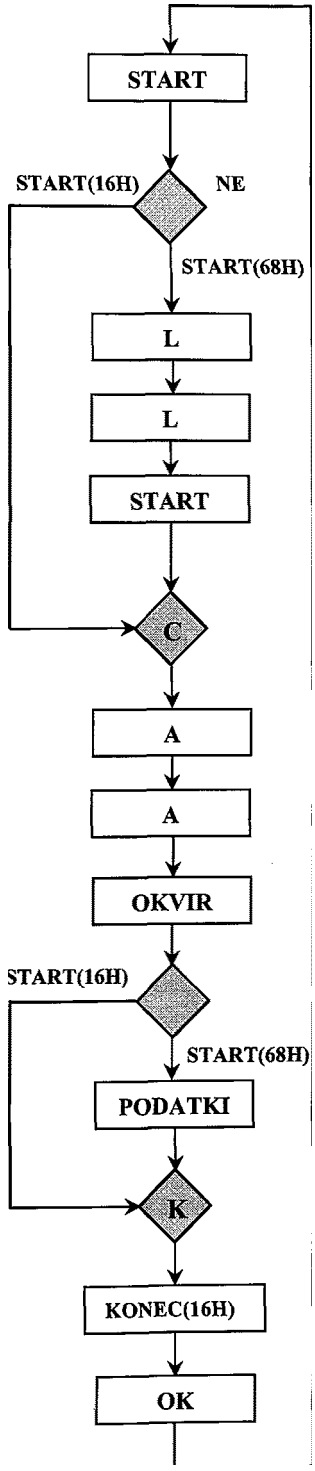
Po uspešnem prenosu kontrolnega zloga pride še zaključni zlog (ta del je enak za oba okvirja), ki algoritem informira o zaključku pošiljanja celotnega podatkovnega okvirja.



Slika 10. Bločni diagram delovanja prevzemnega dela IEC870-5.

5.4.1.2 Oddajna procedura

Pri oddajni proceduri IEC870-5 protokola je situacija bolj enostavna. Ni nam potrebno preverjati oddanih signalov (zlogov), temveč je dovolj, da pošljemo koherentni podatkovni okvir.



Slika 11. Bločni diagram delovanja programske kode oddajnega dela IEC870-5.

Podatkovni okvir se začenja s startnim zlogom 68H v primeru spremenljivega in 10H v primeru nespremenljivega okvirja.

Okvir s spremenljivo strukturo se nadaljuje z dvema zlogoma dolžine in ponovno s startnim zlogom. S kontrolnim zlogom se začne struktura okvirja, ki je enaka obema tipoma okvirjev, nadaljuje pa se z enim ali dvema adresnima zlogoma. Na tem mestu pride znova do razlike med tipoma okvirjev, ker v primeru spremenljivega podatkovnega okvirja pošljemo podatke dolžine L (kontrolni + adresna zloga).

Pošiljanje okvirja (spremenljivega ali nespremenljivega) končamo s končnim zlogom 16H. Tako se procedura zaključi in pri ponovnem pošiljanju podatkovnih okvirjev jo moramo ponoviti.

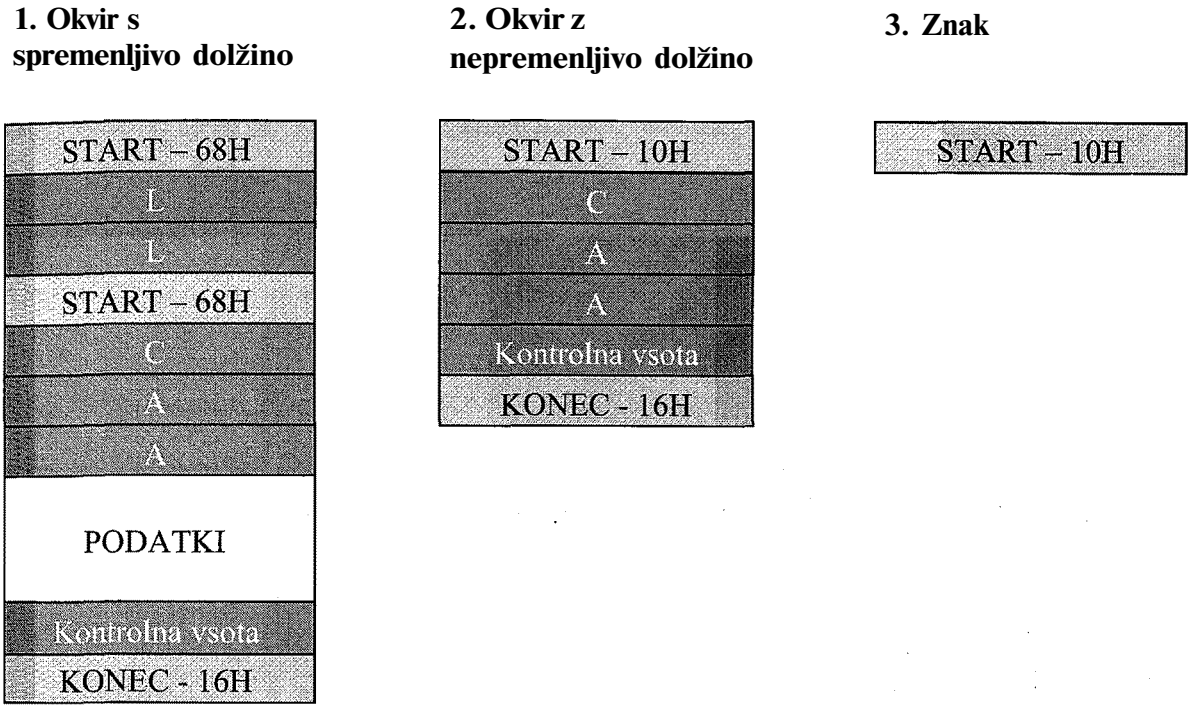
Algoritem se po koncu prejetja okvirja znova postavi na izhodišče in čaka na nov podatkovni okvir. Pri tem algoritem vrne vse potrebne podatke o okvirju aplikaciji, ki nato ustrezno ukrepa naprej po v naprej določenih protokolih. Pomembno je še dodati, da se algoritem izvaja nepretrgoma, ker mora prestreči vse okvirje, ki se pošiljajo po komunikacijskih vodih.

5.4.2 Povezovalni nivo

Povezovalnemu nivoju ustreza IEC870-5-102 komunikacijski protokol. Protokol sprejema in obdeluje le eno povezavo (signal) hkrati v posamezni smer. Vsaka povezava se mora najprej končati (uspešno ali z napako), da se lahko vzpostavi nova.

Pošiljanje podatkov z IEC 870 protokolom poteka preko tako imenovanih podatkovnih okvirjev (data frame). Podatkovni okvir je pravzaprav ogrodje za pošiljanje koristnih informacij prejemniku. Naloga povezovalnega nivoja je ravno sestavljanje takih podatkovnih okvirjev.

V omenjenem protokolu poznamo več tipov podatkovnih okvirjev (*opomba: H pomeni zapis v heksadecimalni kodi):



Slika 12. Nabor podatkovnih okvirjev v IEC870-5-102.

Uporaba posameznega tipa okvirja v komunikaciji zavisi od v naprej določenega in dogovorjenega odziva na določeno zahtevo (povezovalni proceduri se tako reče *"sekvenca")*. Prvi okvir (okvir s spremenljivo dolžino) se uporablja za prenos podatkov, medtem ko ostala dva služita usklajevanju zahtev in prošenj kontrolne in kontrolirane postaje.

5.4.2.1 Okvir s spremenljivo dolžino

Okvir s spremenljivo dolžino sestoji iz začetnega (START) zloga (dolžine enega **byte),** ki prejemniku pove, da se je začel prenos celotnega okvirja. Sledi mu podatek o dolžini celotnega okvirja, ki se po IEC 870 protokolu ponovi dvakrat, ter znova začetni zlog. Naslednji del okvirja pa so že podatki. V podatkovni strukturi je najprej naveden kontrolni zlog, ki vsebuje informacijo o namenu komunikacije. Sledi podatek o naslovu (adresi) prejemnika, ki je lahko velik dva ali en zlog. Za tem pridejo na vrstni uporabniški podatki (podatki, ki jih pošiljamo prejemniku). Celoten okvir se konča s kontrolno vsoto in končnim zlogom. Kontrolna vsota se računa vključno od kontrolnega zloga do vključno zadnjega zloga podatkov.

5.4.2.2 Okvir z nespremenljivo dolžino

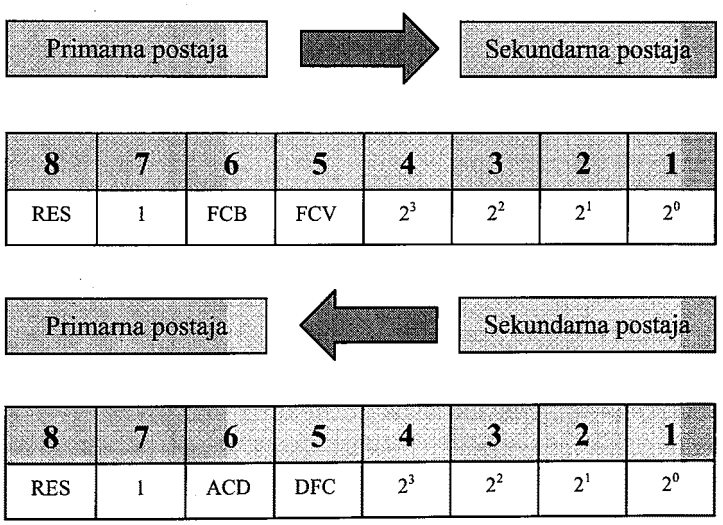
Struktura okvirja z nespremenljivo dolžino je drugačna od okvirja s spremenljivo dolžino. Vsebuje začetni in končni zlog, kontrolno polje, naslovno polje in kontrolno vsoto. Kontrolna vsota vsebuje seštevek kontrolnega zloga ter naslovnega polja.

5.4.2.3 Znak

Tretji okvirje pravzaprav en sam znak (E5-H).

5.4.2.4 Kontrolni zlog

Vsebina kontrolnega zloga zavisi od smeri poteka komunikacije. Slika 13 prikazuje strukturo kontrolnega zloga v obeh smereh komunikacije.



Slika 13. Zgradba kontrolnega zloga v obeh smereh komuniciranja.

RES: Rezerviran bit

FCB: Ob vsakem poslanem podatkovnem okvirju se temu bitu izmenično spreminja vrednost (0 in 1). Na ta način se izognemo podvajanju istih podatkovnih okvirjev, ali pa, v primera izgube okvirja, poskušamo ponovno poslati izgubljeni okvir.

FCV: Bit, ki označuje ali upoštevamo FCB bit ali ne.

Biti od 1. do 4. mesta predstavljajo zlog **funkcijske zahteve** (function call). Funkcijska zahteva nam pove kaj se mora dejansko izvesti, katera procedura, z ustreznimi podatki, ki jih nosi podatkovni okvir.

ACD: Bit nakazuje prisotnost **CLASS** 1 (nekaj več o temu tipu podatkov kasneje) podatkov.
DFC: Nakazuje možnost izgube podatkov ("Data Overflow") pri naslednjem poslanem okvirju.

Funkcijske zahteve:

Pošiljanje funkcijskih zahtev s strani primarne postaje	
Funkcijska zahteva	**Opis zahteve**
0	Ponastavitev povezave
1	Ponastavitev uporabniškega procesa
2	Rezervirano za simetrični tip komunikacije
3	Podatki
4	Podatki
5	Rezervirano
6, 7	Rezervirano za posebno uporabo (po dogovoru)
8	Odziv na to zahtevo definira zahtevo po dostopu
9	Prošnja po stanju povezave
10	Prošnja za zajemanje podatkov tipa "razred 1"
11	Prošnja za zajemanje podatkov tipa "razred 2"
12, 13	Rezervirano
14, 15	Rezervirano za posebno uporabo (po dogovoru)

Tabela 1. Nabor funkcijskih zahtev in pripadajoče zahteve s strani primarne postaje.

Pošiljanje funkcijskih zahtev s strani sekundarne postaje	
Funkcijska zahteva	**Opis zahteve**
0	ACK (Positive Acknowledge): potrditev zahteve
1	NACK (Negative Acknowledge): zavrnitev zahteve, povezava zasedena
2 - 5	Rezervirano
6, 7	Rezervirano za posebno uporabo (po dogovoru)
8	Podatki
9	NACK: zahtevanih podatkov ni na voljo
10	Rezervirano
11	Stanje povezave ali zahteva po dostopu
12	Rezervirano
13	Rezervirano za posebno uporabo (po dogovoru)
14	Povezovalni nivo ne deluje
15	Povezovalni nivo ni implementiran

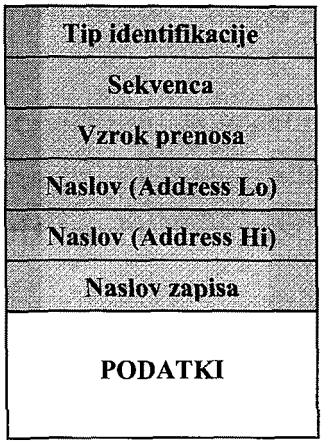
Tabela 2. Nabor funkcijskih zahtev in pripadajoča zahteva s strani sekundarne postaje.

5.4.3 Aplikacijski nivo

Naloga aplikativnega nivoja je zajemanje in obdelovanje želenih podatkov in njihovo urejanje v podatkovne okvirje.

5.4.3.1 Podatki

Osnovna podatkovna struktura, ki jo prenašamo preko IEC870 protokola se imenuje **ASDU (Application Service Data Unit)** oziroma **aplikacijska podatkovna enota.** ASDU je tako srž spremenljivega okvirja kjer se nahajajo podatki, ki jih pošiljamo. Samo strukturo podatkov prikazuje Slika 14.



Slika 14. Struktura ASDU podatkovne enote.

ASDU enota je naprej zgrajena iz parametrov, ki aplikaciji omogočajo razpoznavanje namembnosti podatkov (kakšne podatke vsebuje okvir, kakšen je vzrok pošiljanja podatkov, naslov prejemnika, naslov samega zapisa in seveda same podatke).

Pomembno je še dodati, da so tudi sami podatki znotraj ASDU enote navedeni po specifikacijah, ki natanko določajo vrstni red podatkov. Le na ta način lahko sprejemna enota pravilno rekonstruira prejete podatke.

V naslednjih vrsticah so podrobneje opisani pomeni samih parametrov znotraj ASDU enote.

- Tip identifikacije(Type of identification)

  Tip identifikacije zahteve. Vsakemu ASDU-ju pripada ustrezno število, ki ga imenujemo tip identifikacije zahteve. Preprosto povedano gre za tip ASDU podatkovne enote. ASDU-ji, ki so že v naprej specificirani s protokolom IEC870-5-102 imajo število od 1 do 127. Število 0 se ne uporablja. ASDU-ji z zaporedno številko od 128 do 255 so tako imenovani "custom" ASDU-ji ali naknadno definirani ASDU-ji s strani uporabnikov protokola. Zaželena uporaba ASDU-jev je med 1 in 127. Le na ta način se lahko doseže enakost oziroma univerzalnost protokola med uporabniki.

Razdelitev tipov ASDU-jev:

- Komunikacija v smeri kontrolirana postaja (npr. števec) -> kontrolna postaja (SCADA):

  1... 13, 70.. .72 : Tipi ASDU-jev, ki so definirani v IEC870-5-102 protokolu

  14.. .69, 73.. .99: Možni tipi ASDU-jev definiranih naknadno
- Komunikacija v smeri kontrolna postaja (SCADA) -> kontrolirana postaja (npr. števec):

  100... 123: Tipi ASDU-jev, ki so definirani v IEC870-5-102 protokolu

  124... 127: Možni tipi ASDU-jev definiranih naknadno

t Sekvenca(Variable structure qualifier)

Število informacijskih objektov. Število pove koliko podatkov se nahaja **v** podatkovnem deluASDU-ja.

- **Vzrok prenosa(Cause of transmition)**

Vzrok prenosa podatkov:

- 1.. .47: vzroki definirani v IEC870-5-102 protokolu
- 48… 63: naknadno definirani vzroki prenosa s strani uporabnika

- **Naslov števca (Address Lo)**

Spodnja adresa - spodnjih 8 bitov (Address Lo).

- **Naslov števca (Address Hi)**

Zgornja adresa - zgornjih 8 bitov (AddressHi).

- **Vrsta zapisa (Record Address)**

Vrsta zapisa definira podatkovni tip zapisanega podatka.

Podatkovni tipi:

- 1,11…13,21…23,31…33,41…43,50…56:

  vrste definirane v IEC870-5-102 protokolu
- 2...10, 14...20, 24...30, 34...40, 44...49, 56... 127, 128...255:

  naknadno definirane vrste zapisa prenosa s strani uporabnika

- **Podatki (Data)**

  Podatki, ki jih želimo prenašati. Slednje delimo na več tipov:

  - **"accounting integrated totals":** podatki v zvezi z obračuni (energijski, močnostni registri), izstavo računov itd.
  - **"operational integrated totals":** podatki, ki so v zvezi z bremensko krivuljo (load profile)
  - **"single point information":** dogodki, ki so se zgodili v števcu (ti so lahko predpisani s samim IEC870 protokolom ali pa dogovorjeni s strani proizvajalca)

    "periodically reset": ta vrsta podatkov velja le za prva tipa podatkov in pomeni le relativne vrednosti (delta vrednosti oziroma prirastki vrednosti glede na prejšnje vrednosti); če imamo npr. "periodically reset integrated totals" pomeni, da želimo dostopati do določene relativne vrednosti podatkov v zvezi z obračunom

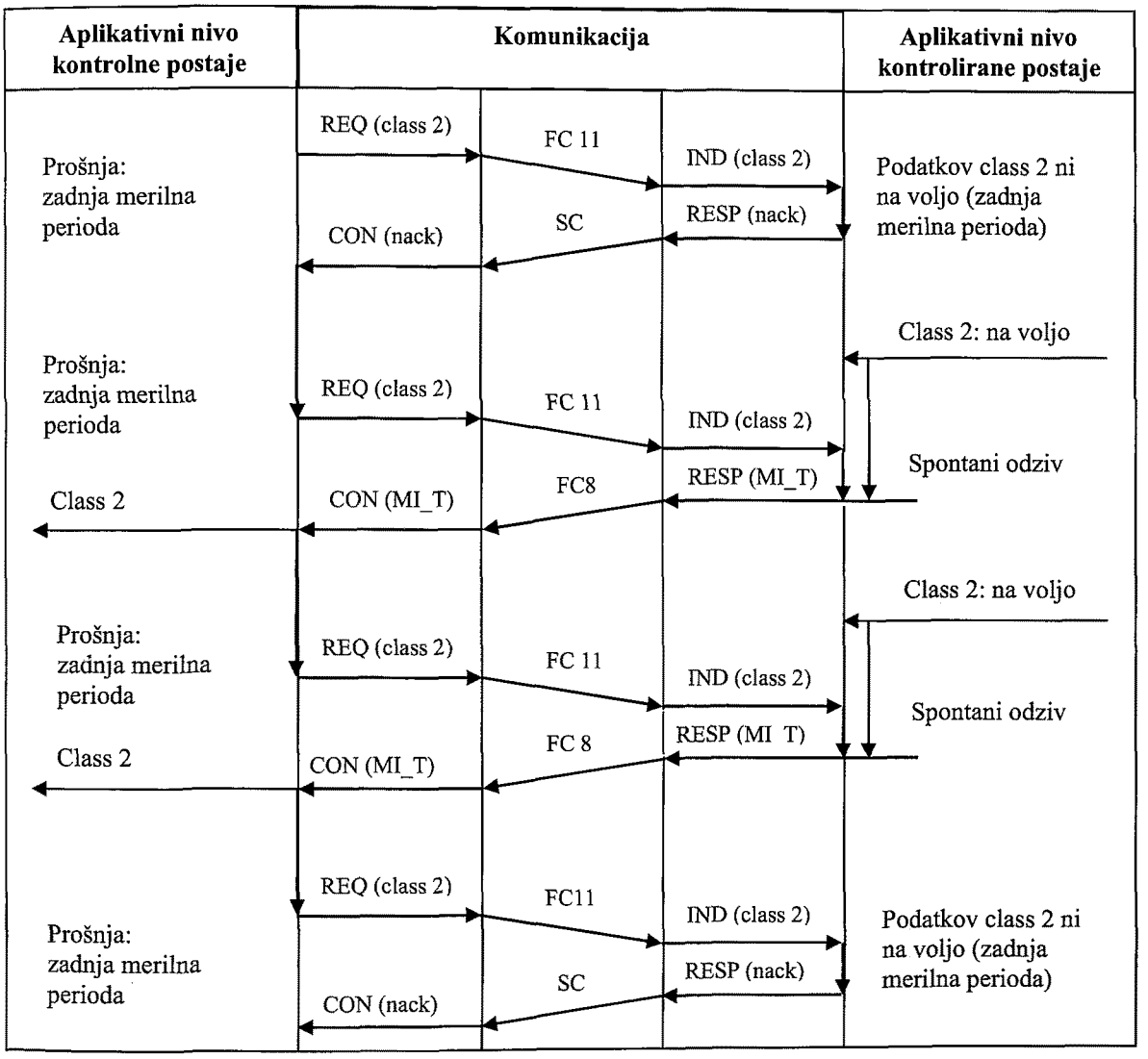
Pomembno je še omeniti, da IEC870-5 in IEC870-5-102 komunikacijski protokol ne podpirata prenosa enot. Kontrolirna naprava mora tako vedeti kakšne enote pošilja kontrolirana naprava in tudi kje se nahaja decimalno mesto.

5.5 Interakcija kontrolirne in kontrolirane postaje

Obe proceduri, sprejemno in oddajno, vodi aplikativni nivo. Zaradi tega mora tudi aplikativni nivo poznati ustrezen odziv na dano zahtevo. Tudi ta del podatkovne izmenjave predpisuje IEC870 protokol. Slike 15, 16 in 17 prikazujejo interakcijo med kontrolno (SCADA npr.) in kontrolirano postajo (elektronski števec električne energije).

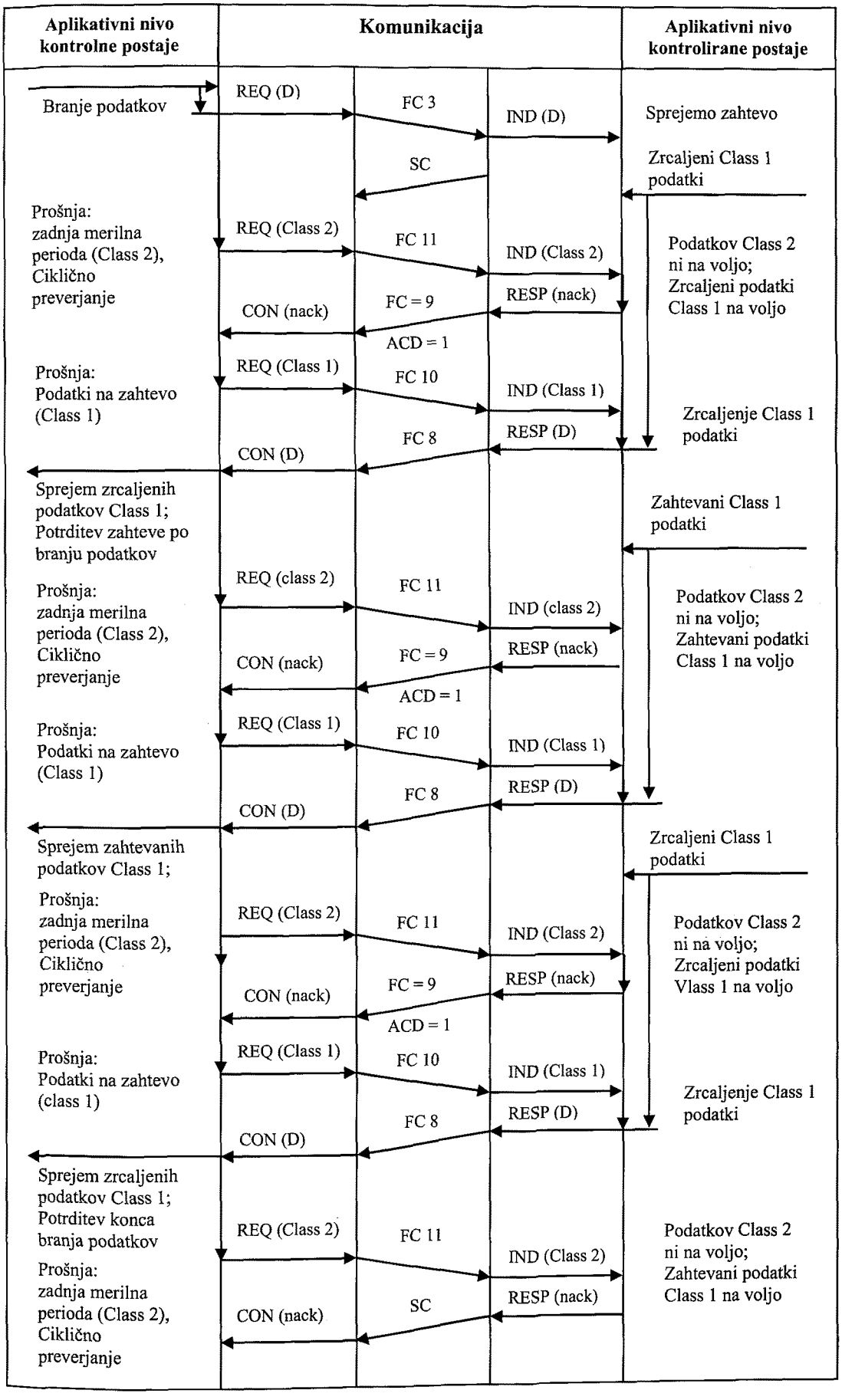
Vse tri slike prikazujejo definirani način odziva obeh strani (kontrolna in kontrolirana postaja). Vidi se, da je kontrolirana postaja vselej v pasivnem stanju, saj vseskozi čaka na zahtevo kontrolne postaje. Po določeni zahtevi oziroma prošnji se kontrolirana postaja odzove z ustreznim odgovorom.

5.5.1 Ciklično zajemanje podatkov



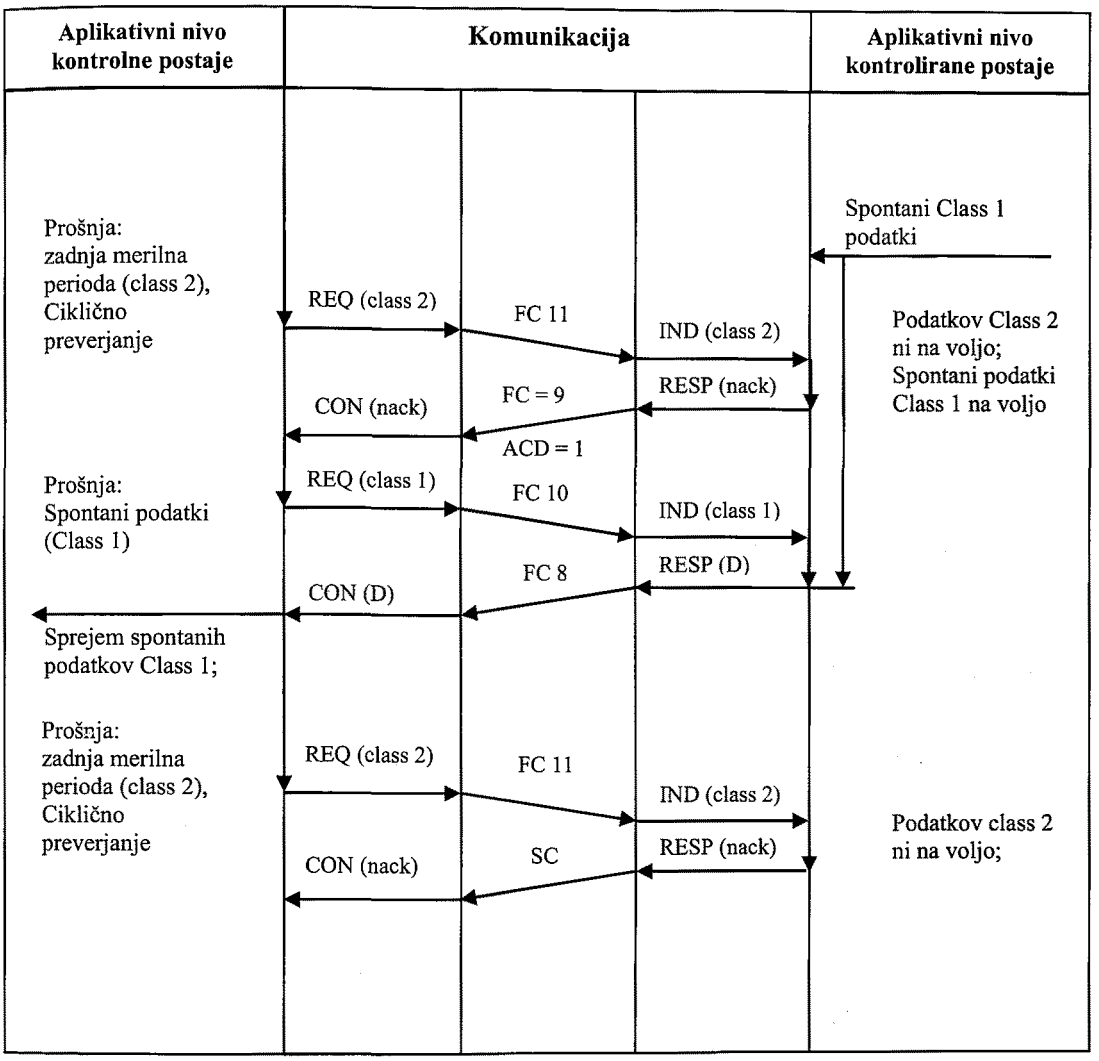
Slika 15. Zakonitosti interakcije med kontrolirno in kontrolirano postajo po IEC870-5 protokolu, pri zajemanju cikličnih podatkov.

5.5.2 Zajemanje podatkov na zahtevo



Slika 16. Zakonitosti interakcije med kontrolirno in kontrolirano postajo po IEC870-5 protokolu, pri zajemanju podatkov na zahtevo.

5.5.3 Zajemanje spontano generiranih podatkov



Slika 17. Zakonitosti interakcije med kontrolirno in kontrolirano postajo po IEC870-5 protokolu, pri zajemanju spontano generiranih podatkov.

Legenda k slikam:

REQ = prošnja (request);
FC = funkcijski klic (Function Call)
SC = posamezen znak (Single Character)
RESP = odgovor (response)
Nack = negativni odgovor
CON = potrdi (Confirm)
IND = zahteva
D = podatki (Data)

Zadnja merilna perioda = splošna oznaka za najnovejše zajete podatke - ti podatki se zajemajo ciklično (na določeno časovno periodo; to so ponavadi informacije o energijah itd.)

6. DLMS/COSEM specifikacija

Na tem mestu je potrebno poudariti, da je DLMS/COSEM specifikacija obsežen in natančno specificiran protokol. Magistrsko delo se bo zato omejilo le na pomembnejše dele specifikacije in se bo močno naslanjalo na magistrsko delo **mag. Aleša Podgornika.** V njegovi magistrski nalogi je specifikacija podrobneje predstavljena, zato za podrobnejše informacije, na željo bralca, priporočam ogled omenjene magistrske naloge[5] ali pa razpoložljivih specifikacij [6][7][8][9] [10].

Po DLMS/COSEM specifikaciji je možno dostopati do podatkov in funkcij v števcu preko standardnega vmesnika, neodvisno od komunikacijskega kanala. V tem primeru je komuniciranje med različnimi števci različnih proizvajalcev zagotovljeno z enim samim gonilnikom, skladnim z DLMS/COSEM specifikacijo [7].
Glavna prednost, ki jo prinaša DLMS/COSEM specifikacija, je neodvisnost uporabljene programske in strojne opreme. To pomeni, da lahko katerikoli sistem za upravljanje komunicira s katerimkoli števcem energije neodvisno od proizvajalca, tipa in komunikacijskega medija.

DLMS (Device Language Message Specification) predstavlja torej sistem sporočil za izmenjavo podatkov in kontrolnih informacij med napravami (aplikacijami implementiranimi na teh napravah) v obliki, ki je neodvisna od uporabljenega komunikacijskega kanala (fizični vodnik) in izvedenih funkcij aplikacije.

COSEM (Companion Specification for Energy Metering) določa pravila za izmenjavo podatkov s števci energije, ki temeljijo na obstoječih standardih. COSEM modelira realno merilno opremo kot niz **logičnih naprav** (logical device), od katerih ima vsaka edinstven identifikator. Informacije znotraj posamezne logične naprave se modelirajo z **vmesniskimi** objekti (interface objects), dostop do teh objektov pa se modelira z **asociacijskim objektom** (association object). Asociacijski objekt nudi informacijo o razpoložljivih sredstvih logične naprave glede na dostopne pravice (teh pravic je več vrst, ločijo pa se po nivoju varnosti asociacije). Informacija, ki jo hranijo vmesniški objekti, je organizirana v **atributih,** katerih

vrednosti predstavljajo lastnosti objekta. Objekt lahko nudi tudi številne metode, s katerimi je možno pregledovati ali spremeniti vrednosti atributov.

Prvi atribut kateregakoli objekta je **logično ime** (logical name), ki je del identifikacije objekta. Logično ime, skupaj z **identifikacijo razreda** (classid) in verzijo, nedvoumno ter neodvisno od proizvajalca določa pomen informacije, ki jo nosi vmesniški objekt. COSEM model omogoča identifikacijo, dostop in interpretacijo informacije v katerem koli števcu energije na neodvisen, nadzorovan in varen način.

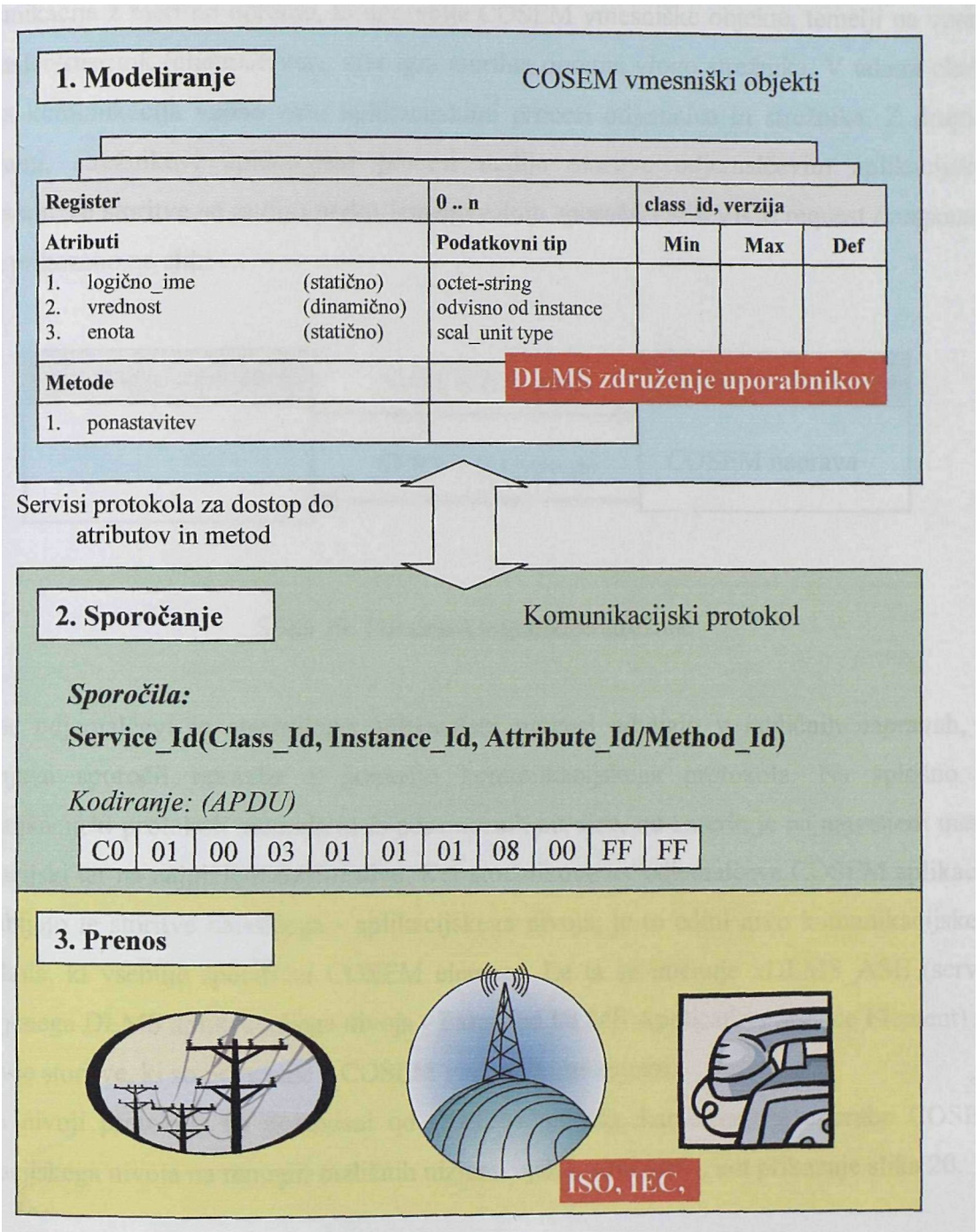
Za dostop do atributov vmesniških objektov skrbijo **xDLMS** (Razširjeni DLMS - Extended DLMS) servisi. To so servisi aplikacijskega nivoja, ki preoblikujejo informacijo v niz bitov. Te se prenašajo preko posameznih nivojev protokola med ciljnima aplikacijama. Nivoji protokola temeljijo na OSI (že prej omenjen) modelu, ki omogoča prenos COSEM podatkovnega modela preko serijskega vmesnika, tokovne zanke, PSTN in GSM modema.

COSEM podatkovni model uporablja splošne gradnike za definicijo kompleksne funkcionalnosti merilne opreme, medtem ko komunikacijski protokol določa dostop in izmenjavo podatkov.

Iz povedanega naredimo krajši povzetek. DLMS/COSEM specifikacija torej ni samo komunikacijski protokol, temveč je tudi sama "filozofija" delovanja števca. Aplikacija, ki vodi delovanje naprave, funkcionira po omenjeni specifikaciji (obdelava podatkov, hranjenje podatkov itd.).

DLMS/COSEM specifikacija sledi trem korakom, ki jih prikazuje slika 18:

- **Korak 1:** model števca in identifikacija podatkov (podatkovni model). Objektni model za opazovanje funkcionalnosti števca s strani uporabnika, ter identifikacijski sistem za vse merjene podatke.
- **Korak 2:** preslikava modela v podatkovne enote protokola PDU (Protocol Data **Unit).** Gre za metodo sporočil za komuniciranje z modelom.
- **Korak 3:** prenos bitov preko komunikacijskih kanalov. Transportiranje podatkov med merilno opremo ter sistemom za zajem podatkov.

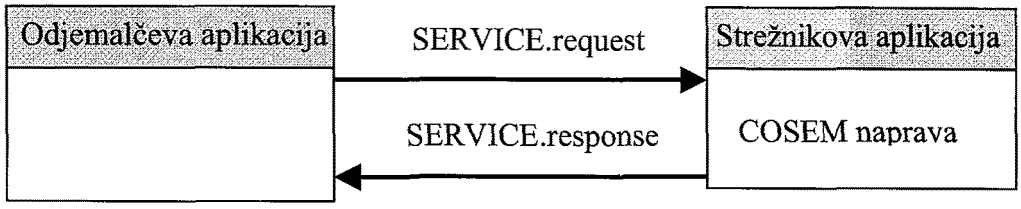


Slika 18. Koraki v COSEM strukturi.

DLMS/COSEM model je evolucija komunikacijske arhitekture, kjer je do sedaj veljalo načelo varčevanja s pasovno širino. Danes je vodilo razvoja poenostavljanje obdelave podatkov in integracija sistemov.

6.1 Zgradba COSEM komunikacije

Komunikacija z merilno opremo, ki uporablja COSEM vmesniške objekte, temelji na vzorcu odjemalec/strežnik (client/server), kjer igra merilna oprema vlogo strežnika. V takem okolju poteka komunikacija vedno med aplikacijskimi procesi odjemalca in strežnika. Z drugimi besedami, strežnikovi aplikacijski procesi nudijo storitve odjemalčevim aplikacijskim procesom. Te storitve se nudijo preko izmenjevalnih sporočil (SERVICE.request /.response), kot je prikazano na sliki….

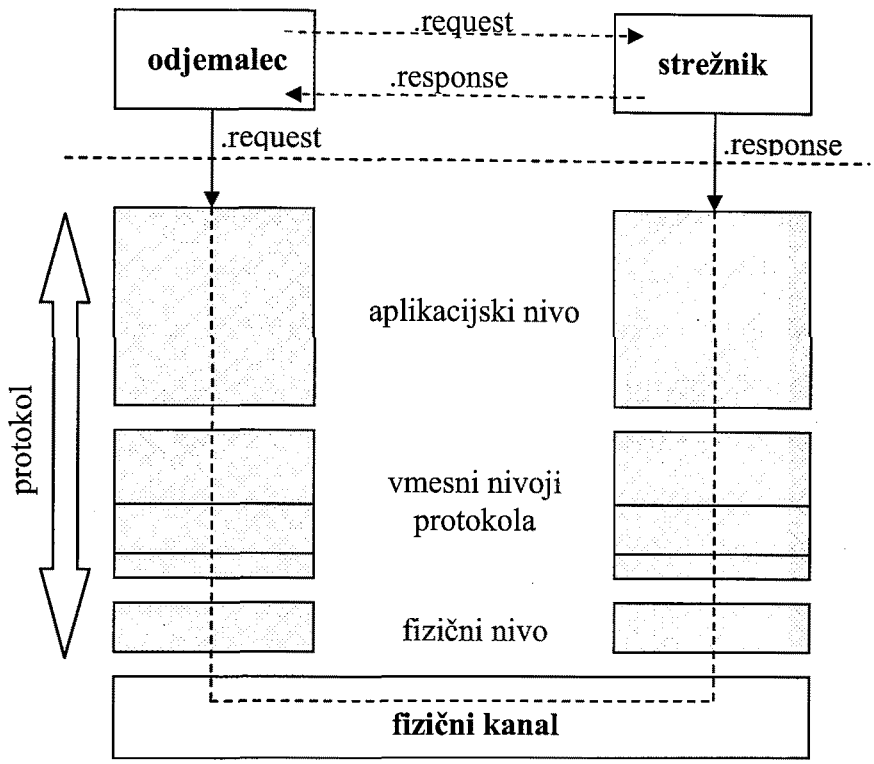


Slika 19. Povezava odjemalec/strežnik.

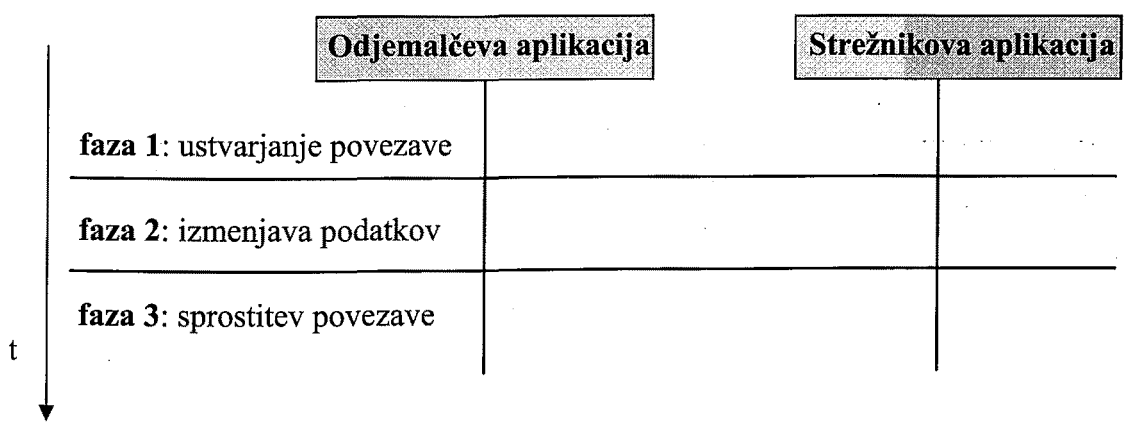
Ker se odjemalčevi in strežnikovi aplikacijski procesi odvijajo v različnih napravah, se izmenjava sporočil opravlja s pomočjo komunikacijskega protokola. Na splošno so komunikacijski protokoli sestavljeni iz posameznih nivojev, od katerih je na najvišjem mestu aplikacijski ter na najnižjem fizični nivo. Ker strežnikove ter odjemalčeve COSEM aplikacije uporabljajo le storitve najvišjega - aplikacijskega nivoja, je to edini nivo komunikacijskega protokola, ki vsebuje specifični COSEM element. Le ta se imenuje xDLMS_ASE (servisi razširjenega DLMS aplikacijskega nivoja - Extended DLMS Application Service Element) ter nudi vse storitve, ki so povezane z COSEM vmesniškimi objekti.
Ostali nivoji protokola so neodvisni od COSEM modela, kar omogoča uporabo COSEM aplikacijskega nivoja na mnogih različnih nižjenivojskih strukturah, kot prikazuje slika 20.

Popolna struktura protokola, ki vključuje tako aplikacijski kot fizični in vse vmesne nivoje se imenuje komunikacijski profil. Značaj komunikacijskega profila določajo vključeni nivoji, njihovi parametri ter tip elementa za nadzor storitev aplikacije ACSE (OSI metoda za vzpostavitev povezave med dvema aplikacijama: Association Control Service Element). ACSE za xDLMS je povezovalno usmerjen protokol, kar pomeni, da lahko aplikacijski

procesi strežnika in odjemalca uporabljajo storitve xDLMS_ASE-ja le, kadar so ti procesi v asociaciji. Zaradi tega poteka komunikacijska seja v treh fazah, kot prikazuje slika 21.



Slika 20. Izmenjava sporočil preko komunikacijskega protokola.



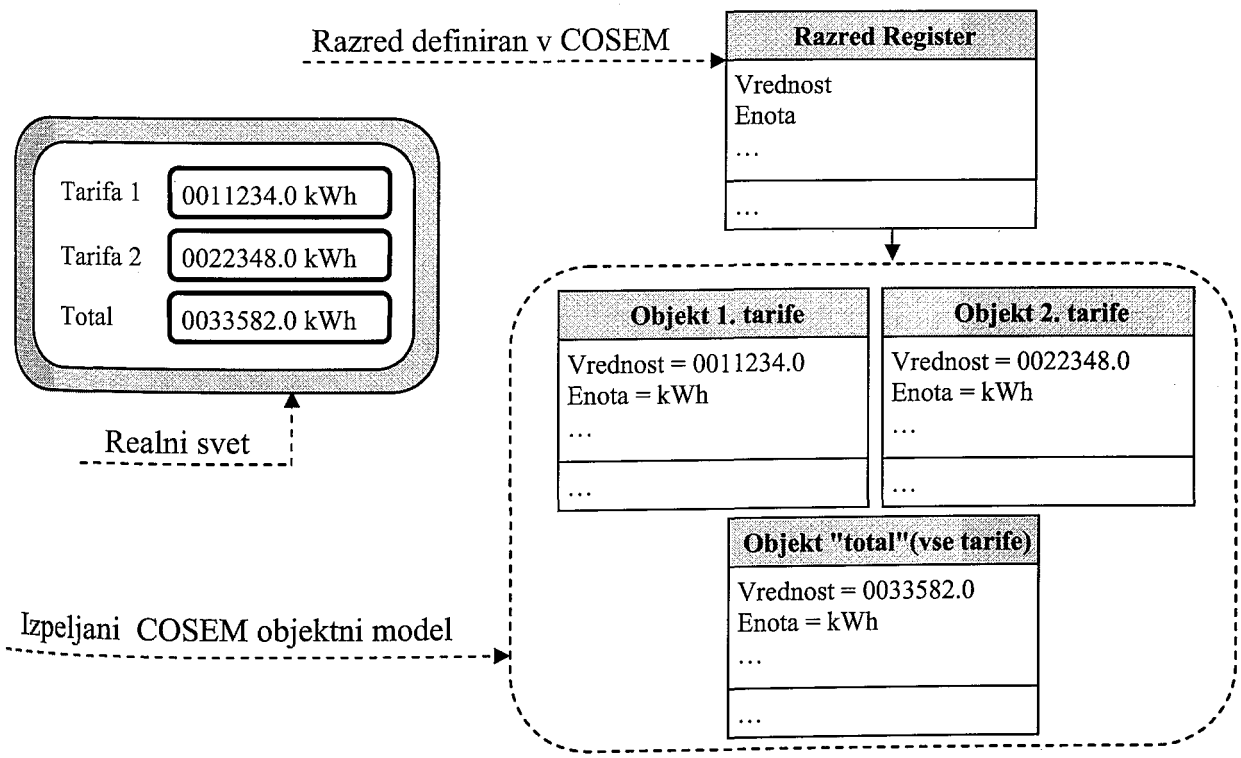
Slika 21. Popolna komunikacijska seja v povezovalno usmerjenem okolju.

Uporaba standardiziranih ACSE storitev zagotavlja izmenljivo storilnost na nivoju aplikacije, medtem ko skupen komunikacijski profil zagotavlja povezljivost. V COSEM modelu začenja

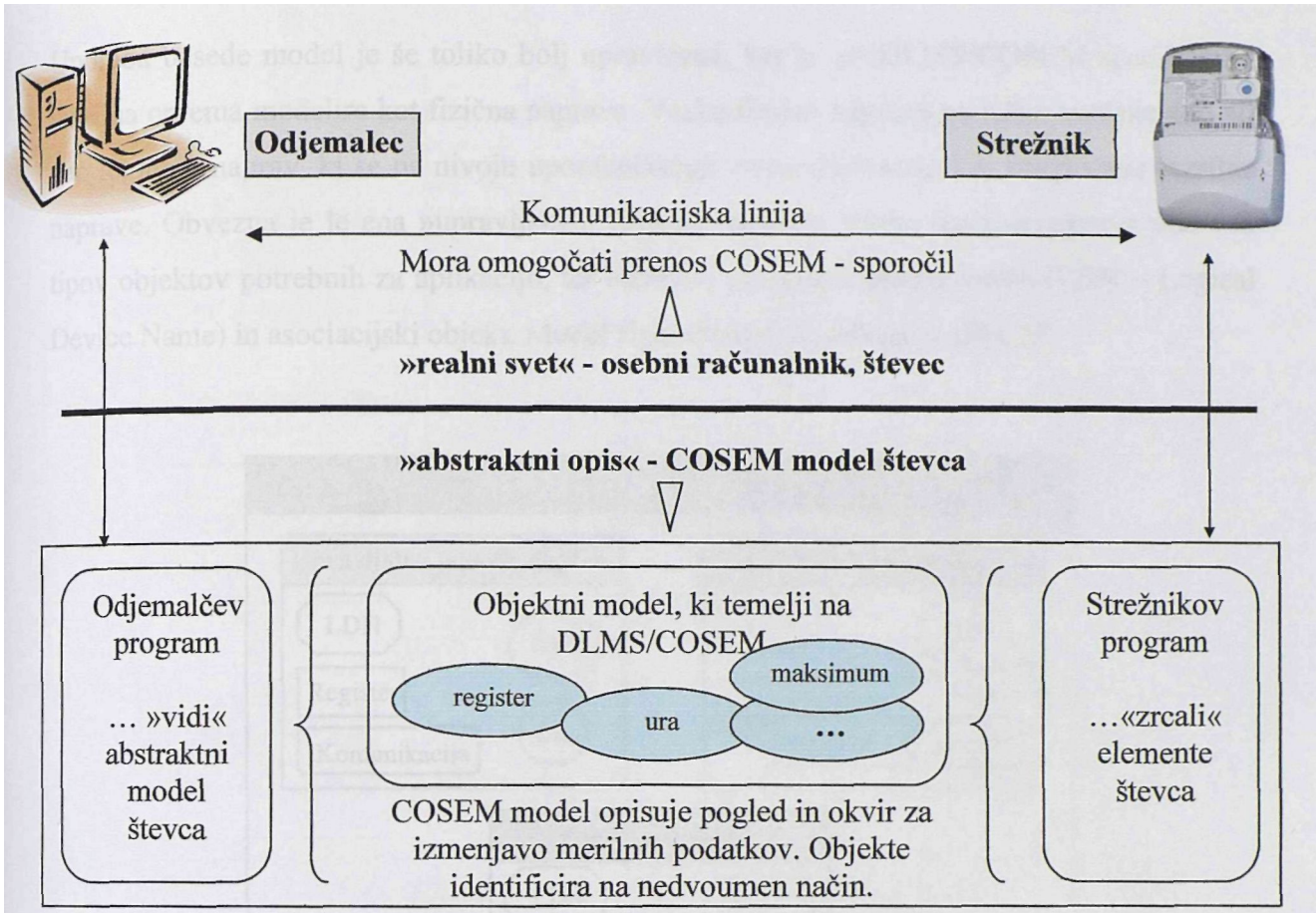
asociacijo med strežnikom in odjemalcem vedno odjemalec. Zaradi možnosti, da odjemalec pri povezovanju z neznano napravo ne pozna implementiranega komunikacijskega profila, je za ta primer v COSEM okolju na voljo posebena storitev za identifikacijo protokola na nivoju aplikacije, ki direktno uporablja storitve fizičnega nivoja protokola.

6.2 COSEM model

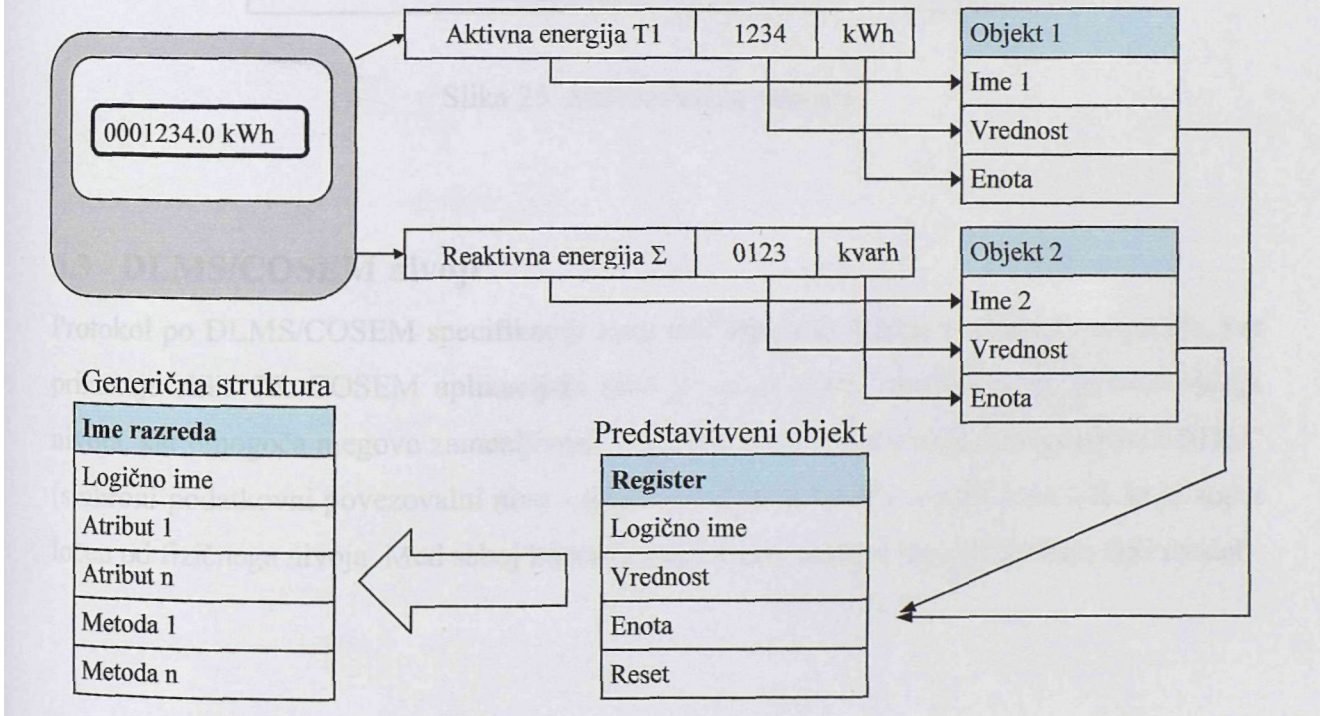
Ideja DLMS/COSEM specifikacije je modeliranje realnega elektronskega števca z modelom, ki ga opisujejo COSEM objekti. Tako se lahko s standardiziranimi deli modelira in predstavi kakršenkoli števec, ki ga lahko razume katerakoli odjemniška COSEM aplikacija. Primer enostavnega števca električne energije s tremi registri ter njegov model prikazuje slika 22. Princip modeliranja po DLMS/COSEM specifikaciji prikazuje slika 23. V modelu se vsi podatki v števcu mapirajo v ustrezne objekte, ki jih določa COSEM. S tem model omogoči funkcionalni pogled na števec. Zaradi enake strukture mnogih podatkov v števcu so podobni objekti kreirani iz istega vmesniškega razreda, vsi vmesniški razredi pa imajo enako generično strukturo. Zaradi tega se lahko do vseh razredov dostopa z enakimi storitvami (GET, SET, ACTION).



Slika 22. Števec in njegov COSEM model.

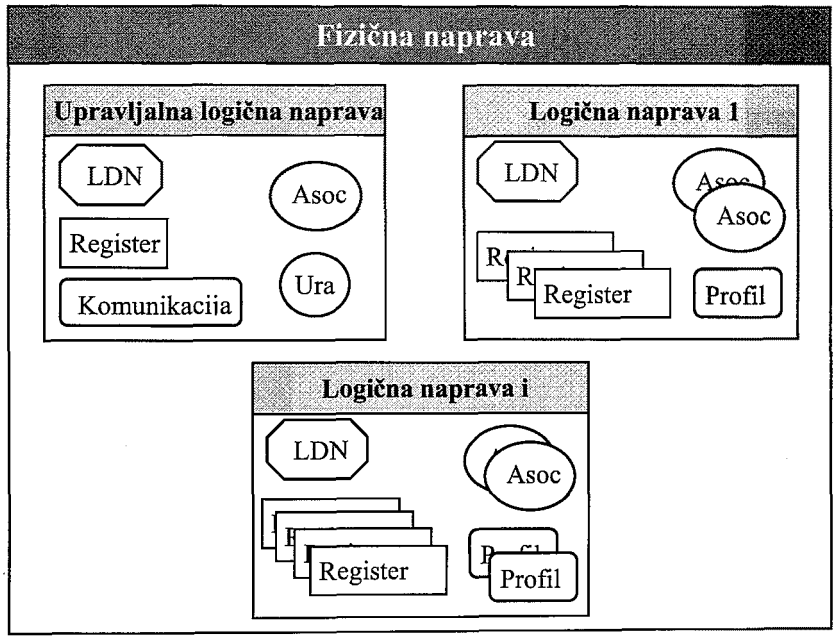


Slika 23. Princip DLMS/COSEM modeliranja.



Slika 24. Mapiranje objektov.

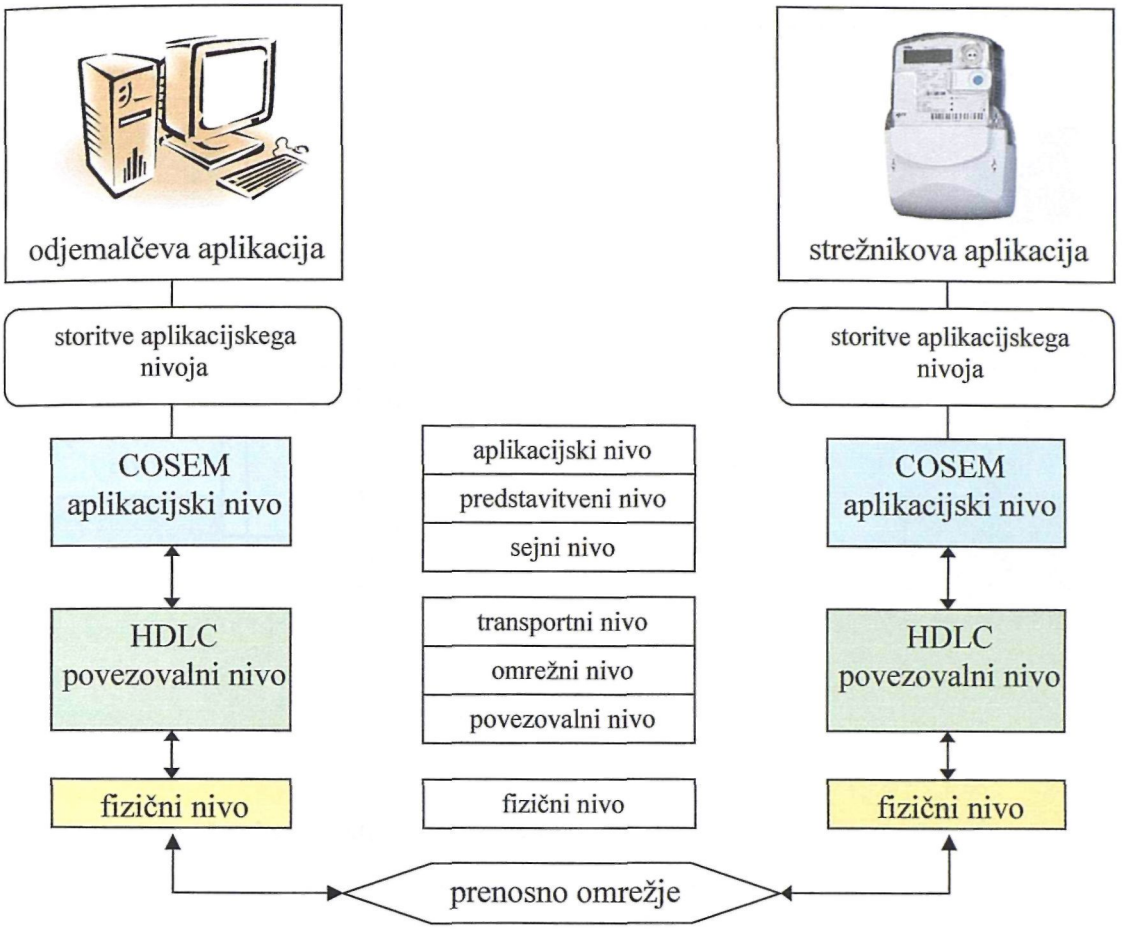
Uporaba besede model je še toliko bolj upravičena, ker se po DLMS/COSEM specifikaciji merilna oprema modelira kot fizična naprava. Vsaka fizična naprava pa lahko vsebuje eno ali več logičnih naprav, ki se na nivoju uporabniškega vmesnika kažejo kot posamezne merilne naprave. Obvezna je le ena »upravljalna« logična naprava. Vsaka logična naprava ima več tipov objektov potrebnih za aplikacijo, ter obvezno »Logično ime naprave« (LDN - Logical Device Name) in asociacijski objekt. Model fizične naprave prikazuje slika 25.



Slika 25. Model fizične naprave.

6.3 DLMS/COSEM nivoji

Protokol po DLMS/COSEM specifikaciji sledi OSI modelu, vendar vsebuje le tri nivoje, kot prikazuje slika 26. COSEM aplikacijski nivo je na ta način neodvisen od povezovalnega nivoja, kar omogoča njegovo zamenljivost. Trenutno je za povezovalni nivo predpisan HDLC (sinkom podatkovni povezovalni nivo - High-Level Data Link Control) protokol, ki je zopet ločen od fizičnega nivoja. Med seboj komunicirajo preko storitev, kot jih definira OSI model.

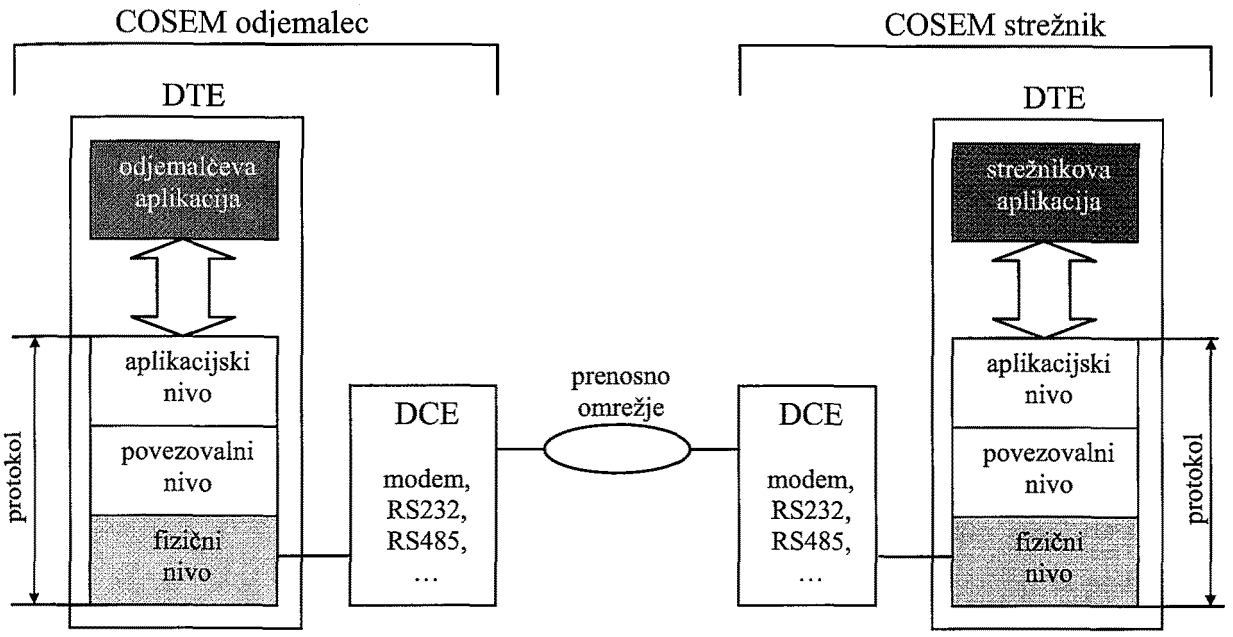


Slika 26. DLMS/COSEM nivoji komunikacijskega protokola.

6.3.1 Fizični nivo

Fizični nivo nastopa kot vmesnik med registrirno napravo (Data Terminal Equipement - DTE) in opremo za prenos podatkov (Data Communication Equipement - DCE).

Z vidika fizičnega nivoja potekajo vse komunikacije v smislu »klicoči sistem« in »klicani sistem«, kjer je »klicoči sistem« tisti, ki določi začetek komunikacije z oddaljenim »klicanim sistemom«. Pomen posameznih sistemov se med celotno sejo komunikacije ne spreminja. Sama komunikacija je razbita na posamezne transakcije, ki se prenašajo od pošiljatelja do prejemnika. Med posameznimi sekvencami transakcije se klicoči in klicani sistem izmenjujeta v vlogi pošiljatelja in prejemnika.



Slika 27. Postavitev fizičnega nivoja.

Fizični nivo omogoča sledeče storitve, ki jih koristijo višji nivoji:

ustvarjanje povezave (request, indication, confirm)

- uničenje povezave (request, indication, confirm)
- prenos podatkov (request, indication)

Poleg naštetih storitev so lahko za višje nivoje potrebne tudi dodatne storitve, ki so del aplikacijskega procesa. Ker so te storitve le lokalnega pomena, jih standard ne predpisuje.

Storitve za ustvarjanje in uničenje povezave koristi direktno aplikacijski proces za fizično povezavo in ne povezovalni nivo. Tako je dopuščena, kot dodatna storitev, možnost identifikacije različnih protokolnih skladov v primeru, ko je na odjemalca priključenih več strežnikov.

Ko je vzpostavljena fizična povezava med odjemalcem in strežnikom, uporablja za prenos podatkov storitvi PH-DATA.request in PH-DATA.indication izključno povezovalni nivo. Kot podatkovna enota fizičnega nivoja PHPDU (Phisical Protocol Data Unit) je definiran en zlog (byte), ki je zaključen s start in stop bitom ter se prenaša od najnižjega proti najvišjemu bitu."

6.3.2 HDLC povezovalni nivo

DLMS/COSEM model omogoča postavitev aplikacijskega nivoja (COSEM) na različne povezovalne nivoje. Trenutno temelji ta nivo na HDLC protokolu, ki je dobro definiran, zato so bila za uporabo v DLMS/COSEM modelu potrebna le nekatera dodatna pravila. Najpomembnejši dodatek je možnost uporabe povezovalne sekvence po IEC62056-21 protokolu, ki je trenutno najširše uporabljen komunikacijski protokol na področju elektronskih števcev električne energije.

HDLC protokol razdeli povezovalni nivo na dva podnivoja, logična kontrola povezave LLC (Logical Link Control), njegova vloga je transparentno nuditi standardne storitve MAC podnivoja, ki skrbi za pravo podatkovno povezavo višjim nivojem, in kontrola dostopa medija MAC (Medium Access Control). Na tem mestu oba podnivoja ne bosta podrobneje opisana, več o njima pa je napisano v literaturi [6].

6.3.3 COSEM aplikacijski nivo

Glavni gradnik aplikacijskega nivoja je COSEM aplikacijski storitveni objekt ASO (Application Servis Object), ki nudi storitve COSEM aplikacijskim procesom, uporablja pa tudi storitve nižjih nivojev protokola.

Tako uporabnikov kot strežnikov COSEM ASO mora vsebovati sledeče obvezne elemente:

asociacije;

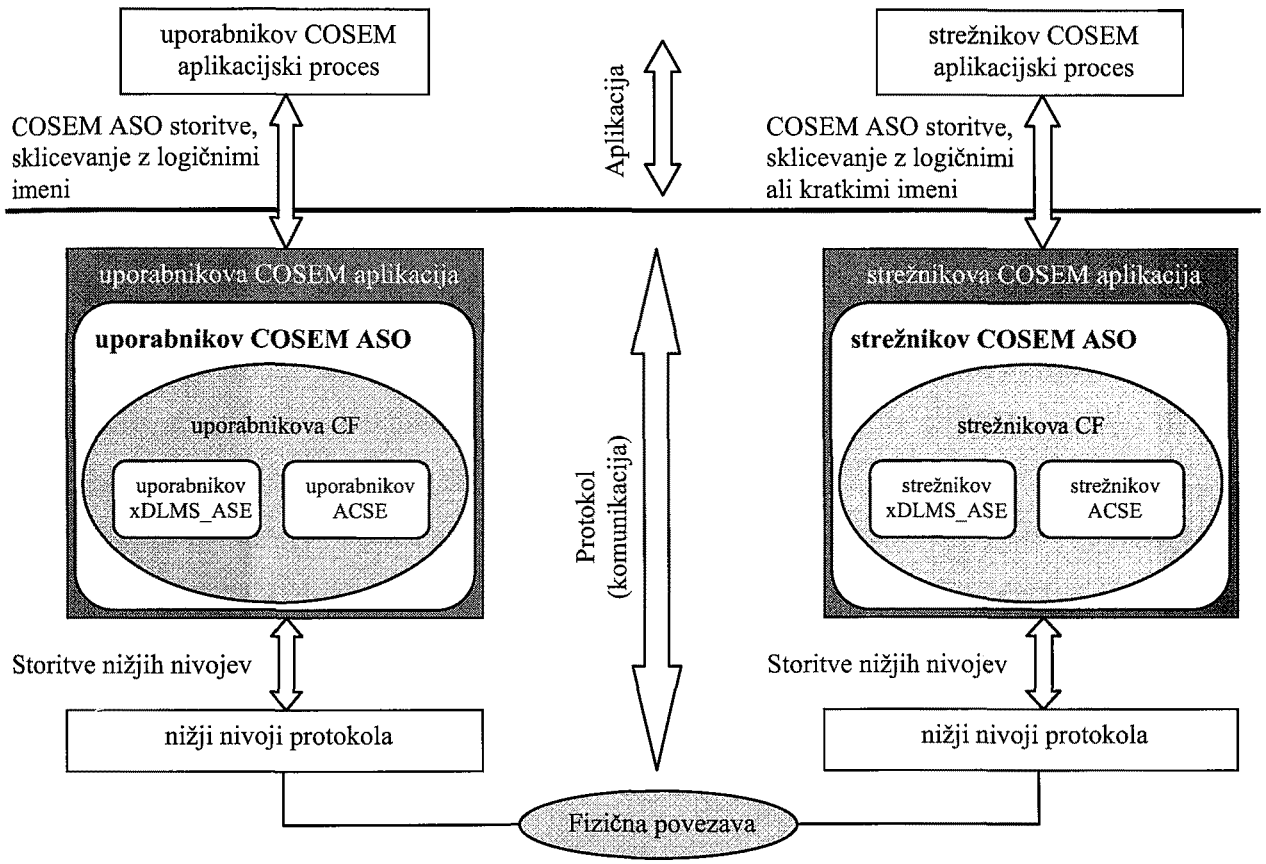
specifični DLMS aplikacijski storitveni element xDLMS_ASE, ki nudi komunikacijske storitve med COSEM opremo;

- nadzorno funkcijo CF (Control Function), ki določa način, na katerega ASO storitve kličejo ustrezne ACSE prototipne storitve, xDLMS_ASE ter storitve nižjih nivojev.

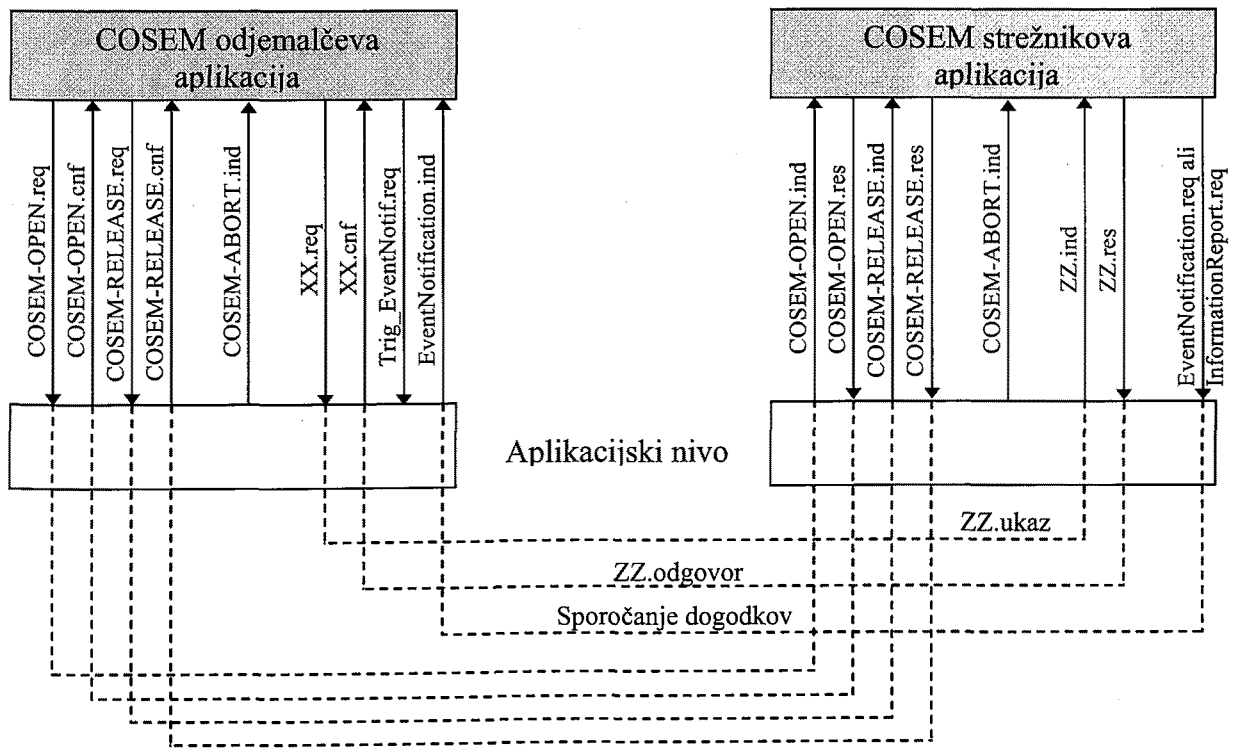
COSEM ASO storitve so razvrščene v tri skupine:

- vzpostavitev ter sprostitev aplikacijske asociacije;
- izmenjava podatkov;
- upravljanje nivojev.

Pregled storitev, ki jih aplikacijski nivo nudi sami aplikaciji prikazuje slika 29.



Slika 28. Struktura COSEM aplikacijskega nivoja.



Slika 29. Storitve COSEM aplikacijskega nivoja.

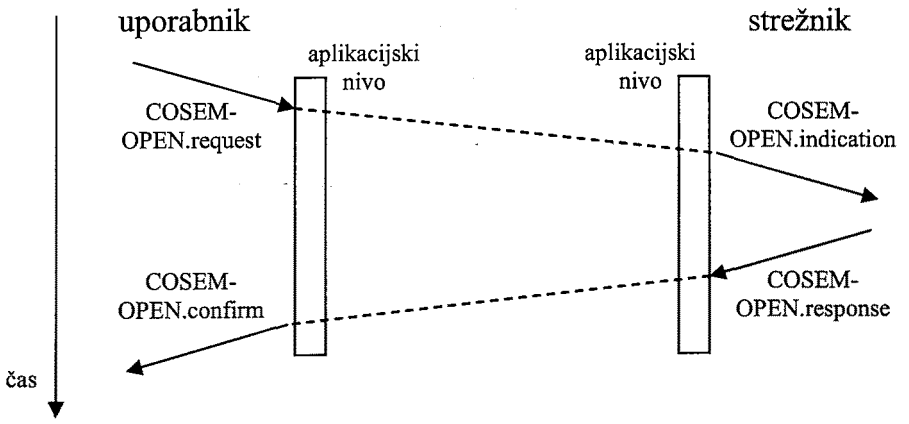
6.3.3.1 Storitve za vzpostavitev ter sprostitev aplikacijske asociacije

Storitve za vzpostavitev ter sprostitev aplikacijske asociacije so sledeče:

- COSEM-OPEN;
- COSEM-RELEASE;
- COSEM-ABORT.

COSEM-OPEN storitev se uporabi v procesu vzpostavitve asociacije ter se nanaša na asociacijske storitve, ki jih nudi ACSE. Ker se v COSEM komunikacijskem profilu asociacija sprosti ali prekine že z odklučitvijo ustreznega nižjega nivoja, se COSEM-RELEASE in COSEM-ABORT storitvi ne nanašata na ACSE.

COSEM-OPEN.request storitev sproži uporabnikova aplikacija za vzpostavitev aplikacijske asociacije s strežnikom. Po povezavi nižjih nivojev se na strani strežnika generira indikacija COSEM-OPEN.indication, ki je preslikava COSEM-OPEN.request storitve uporabnika, na katero strežnik odgovori z COSEM-OPEN.response storitvijo, ki je prenesena na uporabniško stran kot potrditev (COSEM-OPEN.confirm). Tipično povezovalno sekvenco prikazuje slika 30.



Slika 30. COSEM-OPEN storitev.

COSEM-RELEASE storitev se uporablja za "nežno" prekinitev obstoječe asociacije, ki jo lahko zahteva le uporabnik. Sekvenca te storitve je enaka kot pri COSEM-OPEN na sliki 30,

le da se tu uporabi storitev COSEM-RELEASE. COSEM-ABORT storitev se uporabi za indikacijo prekinitve fizične povezave in je enaka za obe strani.

6.3.3.2 Storitve za izmenjavo podatkov

Protokol po DLMS/COSEM specifikaciji definira dva načina sklicevanja za COSEM strežnike: sklicevanje z logičnimi imeni (Logical Names - LN) ter sklicevanje s kratkimi imeni (Short Names - SN). Zaradi tega sta za strežnik definirana dva ločena nabora servisov xDLMSASE. Prvi nabor uporablja izključno LN ter drugi izključno SN sklicevanje, kot prikazuje tabela 3.

	LN	SN
Storitve povezane z **atributi** COSEM vmesniških objektov	GET, SET	Read, Write, Unconfirmed Write
Storitve povezane z **metodami** COSEM vmesniških objektov	ACTION	Write
Storitve za sporočanje dogodkov	Event Notification	Information Report

Tabela 3. COSEM servisi.

V zgornji tabeli naštete storitve se nanašajo na storitve xDLMSASE, katerih večina vsebuje reference na atribute ali metode COSEM vmesniških objektov.

Nabor storitev, ki jih bo uporabljal strežnik med časom komunikacije, je določen z uporabo bloka za preverjanje skladnosti (conformance block), med fazo vzpostavite asociacije. Uporablja se izključno izbrani nabor, ki se med življenjsko dobo aplikacijske asociacije ne sme spremeniti. Uporaba LN ali SN sklicevanja je vezana na asociacijo in ne na strežnik, ki lahko v različnih aplikacijskih asociacijah uporablja različno sklicevanje.

Zaradi zagotovitve transparentnega upravljanja različnih sklicevanj s strani COSEM aplikacijskih procesov, se na strani uporabnika v COSEM aplikacijskem nivoju uporablja le nabor za LN sklicevanje, kar ima dve glavni posledici:

- uporaba edinstvenega in standardiziranega nabora storitev, med COSEM aplikacijskimi procesi ter komunikacijskim protokolom (ti skrijejo posebnosti različnih strežnikov), omogoča določitev programskega vmesnika (API - Application

Programming Interface). Uporaba le teh pa omogoča razvoj uporabniških aplikacij brez znanja o posebnostih posameznih strežnikov;

- kadar COSEM strežnik ne uporablja LN sklicevanja, mora aplikacijski nivo uporabnika vsebovati dodatni element. Namen le tega je preslikava LN nabora storitev, ki ga uporabljajo uporabnikovi aplikacijski procesi, v ali iz nabora storitev, ki ga uporabljajo strežnikovi aplikacijski procesi.

Nabor storitev za izmenjavo podatkov na aplikacijskem nivoju uporabnikove strani vsebuje sledeče storitve (prikazano kot XX na sliki 29):

- GET (.request,, confirm);
  SET (.request,, confirm);
- ACTION (.request,, confirm);

ter nabor storitev za sporočanje dogodkov:

- EventNotification (.indicate);
- TriggerJEventNotificationSending (.request).

Nabor storitev za izmenjavo podatkov na aplikacijskem nivoju strežnikove strani, ki vsebuje LN sklicevanje, vsebuje sledeče storitve (prikazano kot ZZ na sliki 29):

- GET (.indication,, response);
  SET (.indication,, response);
- ACTION (.indication,, response);
- ter storitve za sporočanje dogodkov;
- EventNotification (.request).

V primeru SN sklicevanja na strežnikovi strani se uporabljajo sledeče standardne storitve (definirane v aneksu A standarda IEC 61334-4-41:1996):

- READ (.indication,, response);
- WRITE (.indication,, response);
- UNCONFIRMED WRITE (.indication);
- ter storitev za sporočanje dogodkov;
- InformationReport (.request).

V primeru LN sklicevanja je uporabnikova zahteva (request) identična indikaciji (indication) na strežnikovi strani ter njegov odgovor (response) identičen potrditvi (confirm) na uporabnikovi strani. Informacije posameznih storitev se prenašajo v ustreznih parametrih storitve, ki določajo obliko APDU-ja (Aplikacijska podatkovna enota protokola: Application Protocol Data Unit) za prenos. SN sklicevanje na strani strežnika se v praksi za DLMS/COSEM naprave ne uporablja, zato so v nadaljevanju opisane le storitve z LN sklicevanjem:

- Parametri GET.request (.indication) storitve
- Parametri GET.response (.confirm) storitve
- Parametri SET .request (.indication) storitve
- Parametri SET.response (.confirm) storitve
- Parametri ACTION.request (.indication) storitve
- Parametri ACTION.response (.confirm) storitve
- Parametri Trigger_EventNotification_Sending.
- Parametri EventNotification.indication storitve

6.3.3.3 **Nadzorna funkcija**

Nadzorna funkcija (Control Function - CF) nadzoruje delovanje aplikacijskega nivoja.

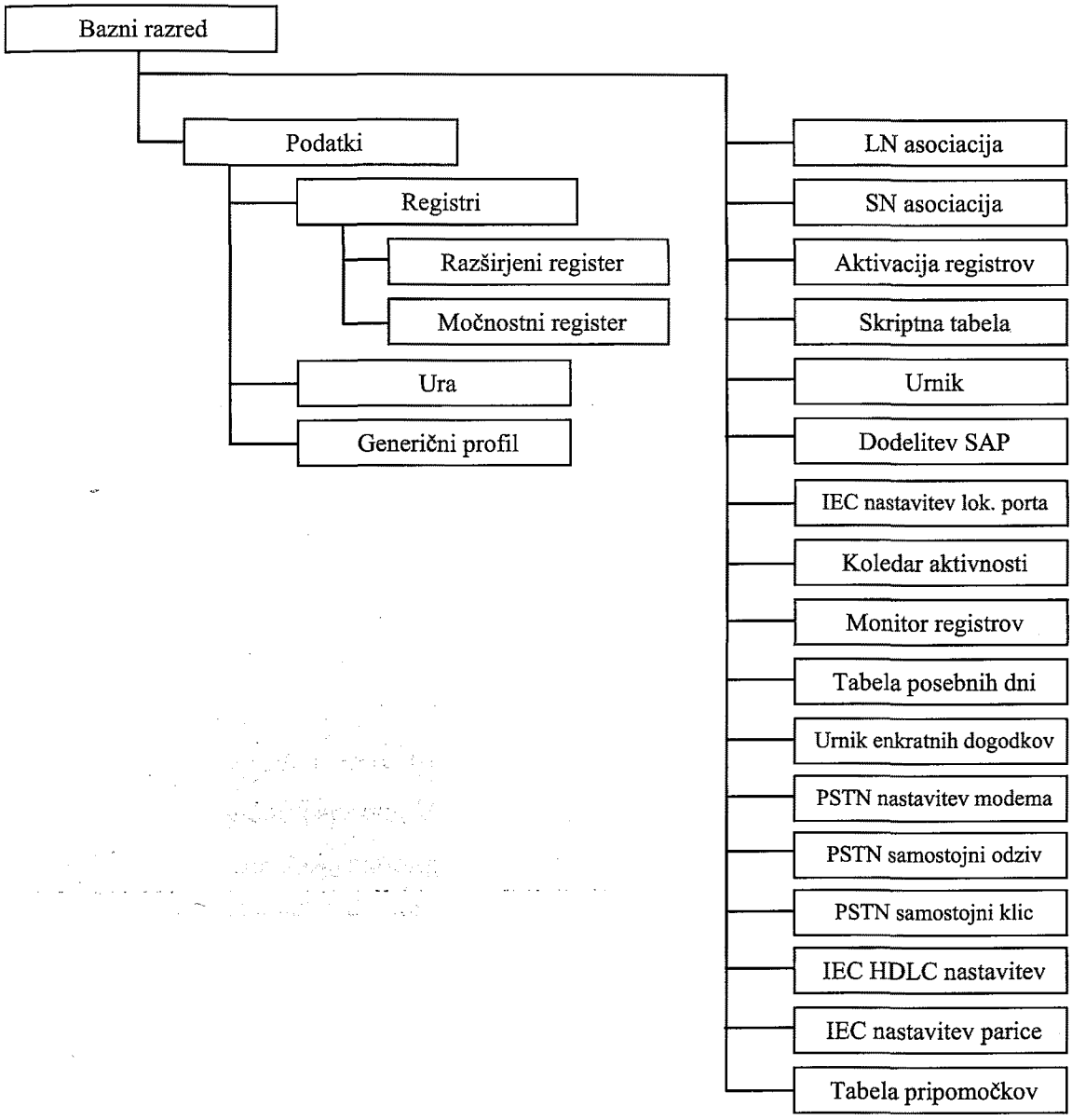
6.4 COSEM razredi

Vmesniški model, ki ga vidi aplikacija, je zgrajen iz definiranih COSEM razredov. Zasnovani so tako, da pokrivajo področje merjenja energije, ki obsega registre, bremensko krivuljo (load profile) rekorder meritev, uro, urnike, aktivacijo registrov ipd. Za nedvoumno identifikacijo posameznih objektov se uporablja OBIS (objektni identifikacijski sistem - OBject Identification System) identifikacijski sistem, kar omogoča bogat nabor standardnih objektov.

Prednost COSEM modela je v tem, da je razširljiv:

- omogoča dodajanje novih verzij razredov;
- omogoča dodajanje povsem novih razredov;
- za podporo posebnosti so dovoljeni specifični razredi, atributi in metode posameznih proizvajalcev.

Specifični razredi so dovoljeni, da se zaščiti univerzalnost standardnih razredov, na katerih popravki s strani proizvajalca niso dovoljeni. Trenutno definirane vmesniške razrede prikazuje slika 33.



Slika 31. Pregled COSEM vmesniških razredov.

6.4.1 Sistem opisa razreda

Vsi razredi sledijo istemu sistemu opisa ter imajo enako strukturo, ki izhaja iz baznega razreda. Bazni razred ni eksplicitno definiran ter vsebuje le atribut **logično_ime,** ki je obvezen v vseh izpeljanih razredih. Sistem opisa vmesniških razredov prikazuje tabela 4.

Ime razreda		Kardinalnost	class_id, VERZIJA		
Atribut(i)		**Podatkovni tip**	**Min.**	**Max.**	**Def.**
1. logično_ime 2. …. 3. ……	(statično) (..) (..)	octet-string …. ….			
Metode		**m/o**			
1. …. 2. ….		…. ….			

Tabela 4. Sistem opisa razredov.

- **Ime razreda**

  Opis razreda (npr. Register, Ura, Urnik)

- **Kardinalnost**

  Določa število instanc razreda znotraj ene logične naprave. Vrednost kardinalnosti se določa z vrednostjo, ki jo lahko omejimo z **minimumom** in **maksimumom** (Razred naj bo instanciran najman **»min.«** krat in največ **»max.«** krat. Pri vrednosti **»min.«** = 0 je razred opcijski, pri »min.«> 0 je instanca razreda obvezena).

- **classid**

  Identifikacijska koda razreda (0 do 65535). Class_id je viden iz »asociacijskega« objekta, ki prikazuje logično napravo. Vrednosti od 0 do 8191 so rezervirane za definicijo s strani DLMS UA (User Association), vrednosti od 8192 do 32767 so rezervirane za razrede posameznih proizvajalcev, od 32768 do 65535 so rezervirane za razrede posameznih uporabniških skupin.

- **Verzija**

  Identifikacijska koda verzije razreda. Verzija je vidna iz »asociacijskega« objekta. Znotraj ene logične naprave morajo biti vse instance nekega razrede iste verzije.

- **Atribut(i)**

  Opisuje atribut(e), ki pripadajo razredu:

  - ***dinamično:*** Vrednost atributa ažurira sama merilna naprava.
  - *statično:* Določa atribut z vrednostjo, ki jo ne ažurira sama merilna naprava (konfiguracijski podatek).

. **logičnoime**

Logično ime je vedno prvi atribut razreda, ki določa instanco (COSEM objekt) razreda ter je obvezen.

. **Podatkovni tip**

Določa podatkovni tip atributa.

. **Min.**

Določa, ali ima atribut minimalno vrednost. Atribut ima lahko poljubno minimalno vrednost x, ali pa je sploh nima.

. Max.

Določa, ali ima atribut maksimalno vrednost. Atribut ima lahko poljubno maksimalno vrednost x, ali pa je sploh nima.

• **Def.**

Določa, ali ima atribut privzeto vrednost. To vrednost ima atribut po resetiranju in inicializaciji. Atribut ima lahko privzeto vrednost x, ali pa je sploh nima.

• **Metode**

Določa spisek metod, ki pripadajo objektu.

• m/o

Definira ali je metoda obvezna ali na izbiro.

6.4.2 Podatkovni tipi

Za razumevanje in pravilno interpretacijo podatkov znotraj objektov morata obe strani, ki komunicirata, poznati podatkovni tip podatka. Za določanje podatkovnega tipa sta na voljo dva mehanizma:

- podatkovni tip je pred-določen v definiciji razreda;
- podatkovni tip določi strežnik, specifično za posamezno instanco razreda, ter ga pošlje s podatkom.

COSEM definira enostavne in kompleksne podatkovne tipe, ki so skupni vsem vmesniškim razredom.

Enostavni podatkovni tipi - podatkovni tipi, ki nosijo le eno podatkovno enoto.	
boolean ***integer*** ***long, double-long*** ***unsigned, long-unsigned*** ***double-long-unsigned***	Enostavni podatkovni tipi, kot jih definira IEC 61334-4-41:1996 Primeri podatkov: integer, integer8 1 byte; long, integer16 2 byte-a; double-long, integer32 4 byte-e.
enum	Elementi enumeracijskega tipa morajo biti definirani v opisu atributa.
real32 ***real64***	Realni podatkovni tipi po REAL specifikaciji IEC 61334-4-41:1996.
visible-string ***octet-string***	Urejen niz ASCII znakov oziroma oktetov (8-bitni bajti).
bit-string	Urejen niz »boolean« vrednosti.

Tabela 5. Enostavni podatkovni tipi [5].

Kompleksni podatkovni tipi - vključenih je več podatkovnih enot iz enostavnih tipov ali pa sama podatkovna enota ni enostavna in predstavlja več različnih informacij.	
Array, compact array	Elementi območja (array) morajo biti definirani v opisu atributa.
structure	Tip strukture mora biti definiran v opisu atributa.
instance specific	Podatkovni tip atributa mora biti specificiran v instanci objekta za določen števec.

Tabela 6. Kompleksni podatkovni tipi [5].

Logično ime ter podatkovni tip zagotavljata nedvoumno interpretacijo podatka, zato lastni podatkovni tipi niso dovoljeni.

6.5 COSEM logična naprava

COSEM logična naprava je množica COSEM objektov, ki modelira realno merilno napravo. Vsaka fizična naprava po DLMS/COSEM specifikaciji mora vsebovati vsaj »upravljalsko logično napravo«, ki je obvezna. Posamezna logična naprava znotraj fizične naprave je edinstveno identificirana z COSEM logičnim imenom naprave, ki je definirano kot niz zlogov *(octet-string)* do vključno 16 oktetov (zlogov). Prvi trije okteti edinstveno identificirajo proizvajalca, za zagotovitev edinstvenosti ostalih oktetov (do 13) pa je odgovoren proizvajalec sam. Za dostop do COSEM objektov v strežniku je potrebno najprej ustvariti aplikacijsko asociacijo. Ta karakterizira vsebino, znotraj katere bosta asociirani aplikaciji komunicirali. Glavni deli te vsebine so:

informacija o sami aplikaciji;

- informacija o COSEM zgradbi;
- informacija o mehanizmih preverjanja pristnosti.

Ta informacija se nahaja v posebnem COSEM »asociacijskem« objektu. Definirana sta dva tipa asociacijskega objekta. Eden uporablja SN sklicevanje, drugi pa LN sklicevanje.

Glede na vzpostavljeno asociacijo med odjemalcem in strežnikom lahko strežnik odobri različne pravice dostopa. Pravice dostopa se nanašajo na niz »vidnih« objektov, ki so dostopni (vidni) znotraj dane asociacije. Dostop do atributov in metod v teh objektih je lahko še dodatno omejen z asociacijo.

Spisek vidnih COSEM objektov je dostopen s strani uporabnika preko liste objektov (object list) atributa ustreznega asociacij skega objekta (LN ali SN sklicevanje), ki mora biti, poleg objekta z imenom logične naprave, viden v vseh aplikacijskih asociacijah.

6.6 Postopki preverjanja pristnosti

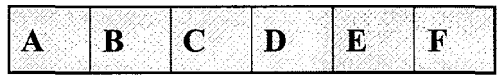
Za določeno zaščito pred neželenim dostopom do naprave (števca) skrbi preverjanje pristnosti, ki lahko poteka na več nivojih. To pomeni, da imamo lahko več stopenj preverjanja:

- **Preverjanje pristnosti nizke varnostne stopnje LLS (Low Level Security)** se običajno uporablja pri komunikaciji preko kanalov, ki nudijo zadovoljivo preprečevanje "prisluškovanja".
- **Preverjanje pristnosti visoke varnostne stopnje HLS (High Level Security)** se običajno uporablja pri komunikaciji preko kanalov, ki ne nudijo prave zaščite pred

"prisluškovanjem". V tem primeru je predviden 4-prehodni protokol preverjanja pristnosti[6].

6.7 OBIS (objektni identifikacijski sistem)

OBIS identifikacijski sistem služi kot osnova za COSEM logična imena. Glavni namen OBIS sistema je zagotoviti nedvoumno identifikacijo podatkov, ki so na voljo za določeno asociacijo. Sestavljen je iz šestih skupin v hierarhični strukturi. Pomembno je še omeniti, da OBIS identifikacija ni namenjena zgolj DLMS/COSEM protokolu. Gre za univerzalen identifikacijski sistem, ki se lahko uporablja kjerkoli (uporablja se recimo tudi v IEC62056-21 komunikacijskemu protokolu). OBIS strukturo prikazuje Slika 34.



Slika 32. OBIS struktura.

- **Skupina A**

  Definira značilnost predstavljenega podatka. Območje vrednosti je med 0 in 15. (npr: 0: Abstraktni objekt; 1: Objekt, povezan z elektriko; itd.)

- **Skupina B**

  Definira številko kanala. Ta skupina definira številko vhodnega kanala v primeru merilne naprave z več vhodi (podatkovni koncentrator itd.), za iste ali različne tipe energije. Tako so lahko identificirani podatki iz različnih virov. Definicije za to skupino so neodvisne od skupine A. Območje vrednosti je med 0 in 255

- **Skupina C**

  Definira abstraktne ali fizične podatke glede na vir informacije (tok, napetost, moč, temperatura itd.) Definicija je odvisna od vrednosti skupine A. Meritev, tarifno procesiranje in hranjenje teh podatkov definirajo vrednosti skupin D, E in F. Območje vrednosti je med 0 in 255. (npr: 0: splošni objekt; 1: ΣL_i; Aktivna moč+ itd.).

- **Skupina D**

  Definira tipe ali rezultate procesiranja fizičnih količin, ki so povezane s skupinama A in C. Območje vrednosti je med 0 in 255. (npr: 0: Povprečje merilne periode; 1: Skupni minimum litd). V primeru, da je vrednost skupine C enaka 94 (A = 1),

predstavlja skupina D nacionalne specifične sheme, ki jih določata skupini E in F. (npr: 0: Finska; 1: ZDA itd.)

- **Skupina E**

  Tu se dodatno definirajo vrednosti merilnih rezultatov določenih s skupinami A, B, C in D. Pri električnih veličinah (A=1) se tu določa tarife. Območje vrednosti je med 0 in 255. (npr: 0: skupna tarifa; 1: tarifa 1 itd.).

. **Skupina F**

  Definira hranjenje podatkov (preteklih vrednosti), ki jih določajo skupine A do E. Če to pri podatku ni pomembno, se lahko skupina uporabi za dodatno klasificiranje.

6.8 Preizkušanje DLMS/COSEM skladnosti

Zaradi zagotavljanja združljivosti DLMS/COSEM naprav naj bi vse naprave uspešno opravile test skladnosti. Test skladnosti zagotavlja pravilno implementacijo standardnih tipov objektov in storitev. Preizkušanje združljivosti mora zajeti:

- preizkušanje vseh objektov in storitev, ki so implementirane **(pozitivni test);**
- preverjanje pričakovanih odgovorov v primeru napačnih ali neimplementiranih storitev in objektov **(negativni test).**

Za preizkušanje se uporabljajo orodja za preizkušanje skladnosti CTT (Conformance Test Tool), ki omogočajo testiranje implementacije DLMS/COSEM specifikacije v COSEM strežniku. Ker je poudarek na pravilni implementaciji COSEM zahtev, je to preizkušanje skladnosti. Uspešno izveden test povečuje verjetnost, da bo naprava združljiva z DLMS/COSEM uporabnikom, ki ustreza specifikaciji. CTT se uporablja le za preizkušanje SDLMS/COSEM strežnikov in mora izpolniti sledeče zahteve:

- preizkus mora pokriti vse standardne objekte, atribute in metode;
- preizkus je omejen na komunikacijo preko optične sonde, tokovne zanke ali RS232 vrat;
- preizkus vseh definiranih primerov napak, na katere se mora strežnik odzvati na način, definiran v standardu;
- izvedba preizkusa s CTT mora biti čim bolj samostojna, z minimalnim posegom operaterja (standardiziran preizkus);

- posamezne preizkuse naj nadzira preizkusna procedura, ki mora biti zaščitena pred prilagoditvami s strani preizkusnega operaterja. Preizkusna procedura mora biti v obliki, kije razumljiva tako človeku kot preizkusni napravi (avtomatizacija);
- podnabor preizkusnih procedur je definiran z izjavo o skladnosti implementacije protokola PICS (Protocol Implementation Conformance Statement) ter dodatnimi informacijami za testiranje implementiranega protokola PIXIT (Protocol Implementation eXtra Information for Testing);
- rezultat CTT-ja je preizkusno poročilo, ki vsebuje seznam opravljenih preizkusov ter ustreznih ugotovitev (opravil, napaka, izpuščen). Preizkusno poročilo mora vsebovati nedvoumno definicijo preizkušanega strežnika ter ustrezne dele PICS in PIXIT, ki ju mora zagotoviti proizvajalec;
- CTT je omejen na preizkušanje strežnika na nivoju komunikacijskega vmesnika. Ostale funkcije strežnika v preizkusu niso zajete;
- vmesniški razredi, definirani s strani proizvajalca, niso zajeti v preizkus;
  standarni razredi, nadgrajeni s strani proizvajalca, se preizkušajo v okviru zmožnosti.

PICS

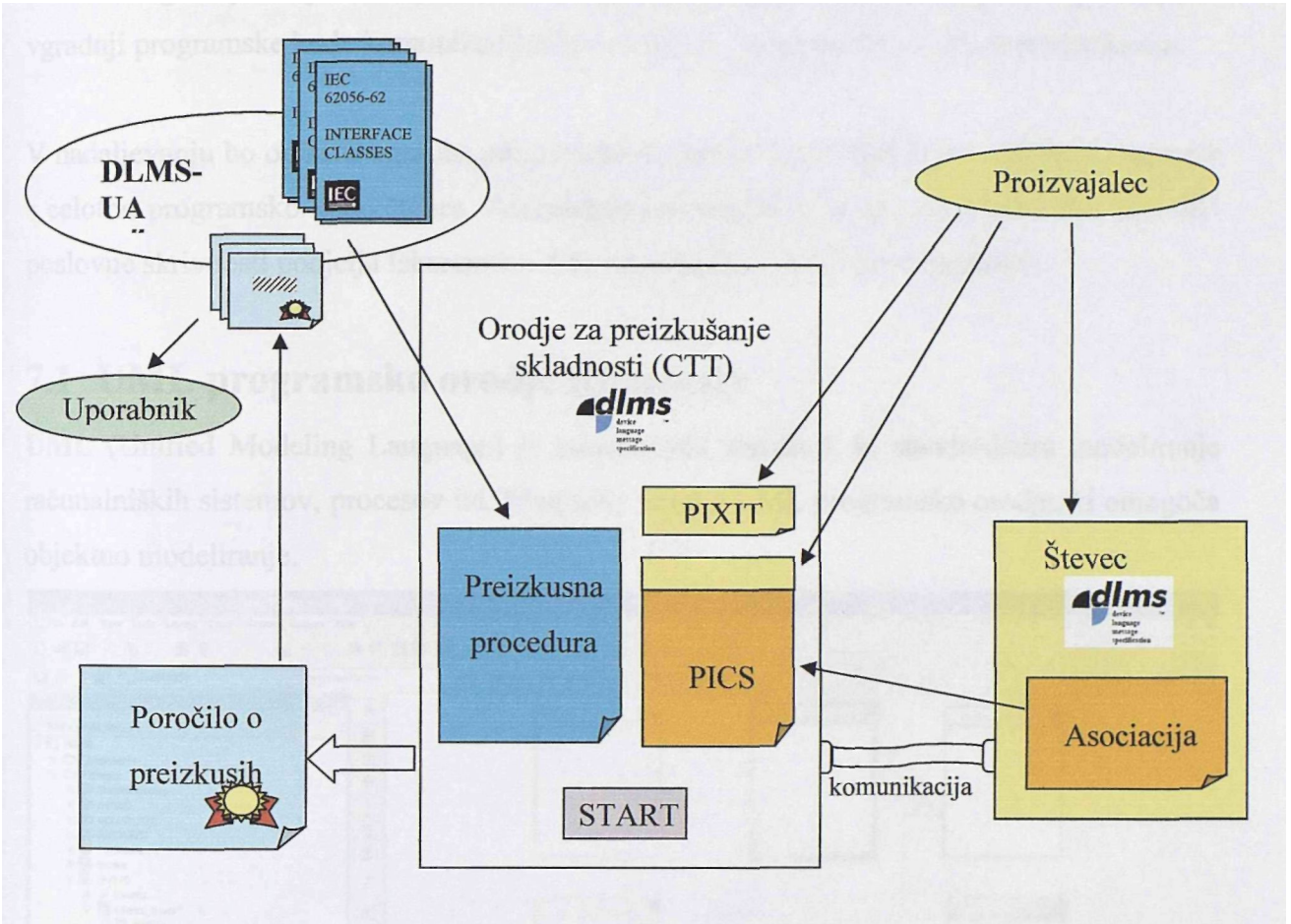
Vse naprave, ki ustrezajo DLMS/COSEM specifikaciji, morajo imeti izjavo o skladnosti implementacije protokola (PICS), ki identificira vse implementirane dele DLMS/COSEM protokola. PICS je pisan dokument, tega izdela proizvajalec naprave, ki identificira vse implementirane opcije v napravi. DLMS/COSEM PICS se šteje kot javni dokument, dostopen vsem zainteresiranim strankam.

PIXIT

PIXIT je dokument, ki specificira vse potrebne informacije (in jih ne vsebuje PICS) za avtomatično testiranje ter ga pripravi proizvajalec.

Glavni cilj preizkušanja DLMS/COSEM skladnosti s CTT je avtomatizacija preizkusnih procesov. Orodje obdeluje, interpretira in izvaja serije preizkusnih procedur ter kot končni produkt vrne rezultat preizkusov v obliki poročila. Preizkusne procedure izdaja DLMS-UA in jih končni uporabnik orodij ne more spreminjati. CTT mora biti sposoben na osnovi PIXIT

dokumenta ustrezno oblikovati avtomatično preizkusno proceduro. Osnovni princip preizkušanja s CTT prikazuje slika 33.



Slika 33. Diagram poteka preizkušanja skladnosti.

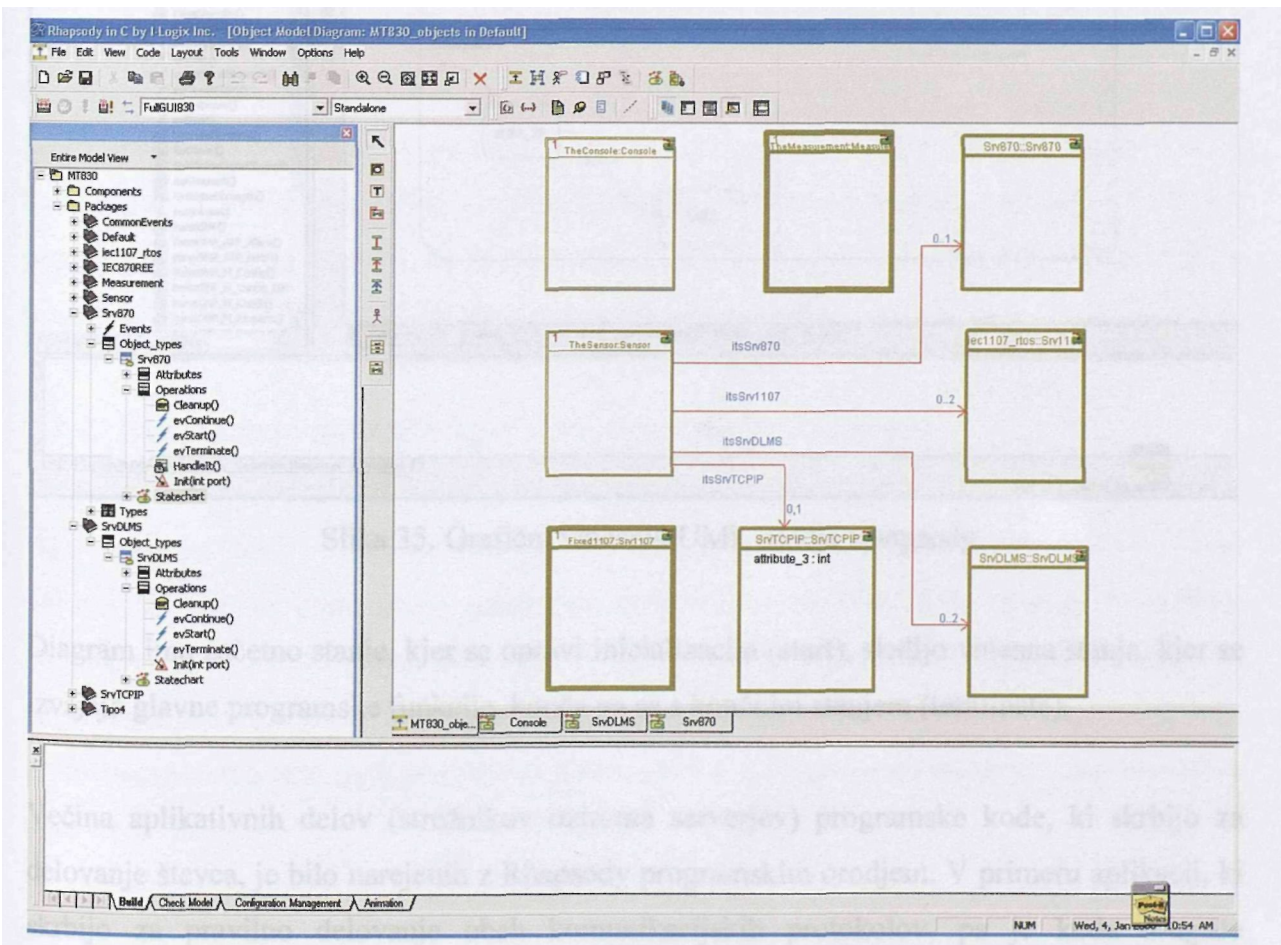
7. Vgradnja modulov v elektronski števec in testiranje

Elektronski števci (MT830, MT831, MT860) v katere sta bila vgrajena oba komunikacijska protokola, so zgrajeni modularno (programsko kot tudi strojno). Zaradi modularnosti je tako potekala vgradnja lažje. Ko govorimo o vgradnji komunikacijskih modulov, govorimo o vgradnji programske kode komunikacijskih modulov v programsko kodo celotnega števca.

V nadaljevanju bo opisana zgradba programske kode komunikacijskih protokolov v povezavi s celotno programsko kodo števca. Podrobneje izvorna koda ne bo analizirana, ker je to del poslovne skrivnosti podjetja Iskaremeco d.d., navsezadnje pa to tudi ni potrebno.

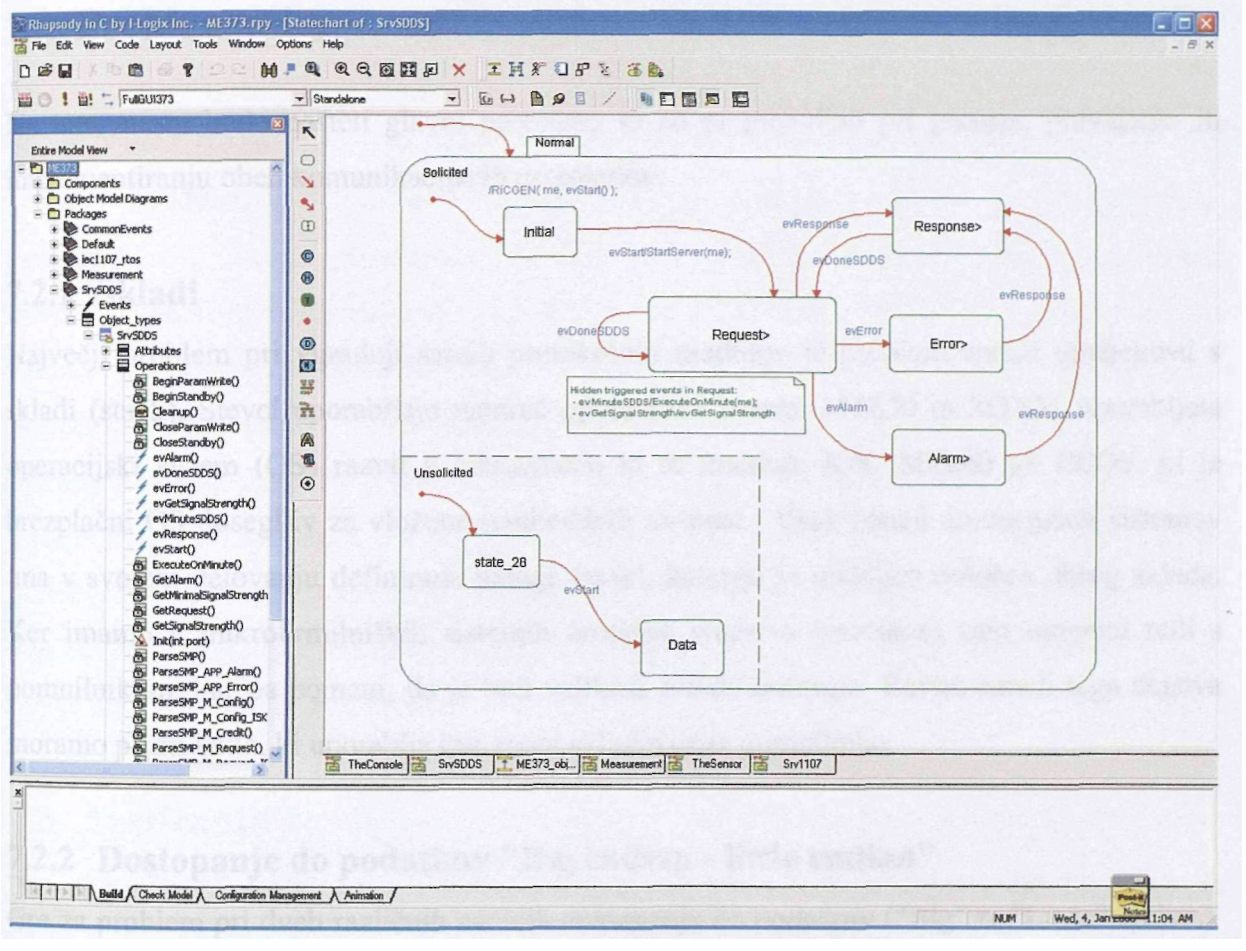
7.1 UML programsko orodje Rhapsody

UML (Unified Modeling Language) je računalniški standard, ki standardizira modeliranje računalniških sistemov, procesov itd. Rhapsody je tako UML programsko orodje, ki omogoča objektno modeliranje.



Slika 34. Definiranje objektov z UML orodjem Rhapsody.

Na ta način se močno olajša generiranje programske kode za vložene (embedded) sisteme. Najprej se z orodjem definirajo objekti in njihove medsebojne relacije (slika 34). Z grafičnim vmesnikom, preko katerega generiramo diagram prehajanja stanj za vsak posamezen objekt, pa definiramo funkcionalnost posameznega objekta. Na ta način nam Rhapsody generira programsko kodo, ki se vključi v želeni projekt (programska koda števca). Slika 35 prikazuje vmesniško okno preko katerega se nariše diagram prehajanja stanj.



Slika 35. Grafični vmesnik UML orodja Rhapsody.

Diagram ima začetno stanje, kjer se opravi inicializacija (start), sledijo vmesna stanja, kjer se izvajajo glavne programske funkcije, konča pa se s končnim stanjem (terminate).

Večina aplikativnih delov (strežnikov oziroma serverjev) programske kode, ki skrbijo za delovanje števca, je bilo narejenih z Rhapsody programskim orodjem. V primeru aplikacij, ki skrbijo za pravilno delovanje obeh komunikacijskih protokolov, pa je koda drugače zasnovana. Na najvišjem nivoju je preprost strežnik (server; ta je bil zgrajen s pomočjo Rhapsody orodja), ki skrbi za periodično klicanje funkcij, inicializiranje in zaključevanje

aplikativnega dela protokolov. Te funkcije so implementirane ločeno od strežnikove aplikacije (druga »C« datoteka).

V Rhapsody-ju je vsak aplikacijski proces definiran s svojim objektom. V števcu so to: konzola (LCD itd.), senzor, merilni del ter komunikacija (IEC62056-21, IEC870-5-102, DLMS/COSEM itd.).

7.2 Problematika

Na tem mestu bodo našteti glavni problemi, ki so se pojavljali pri pisanju, prevajanju in implementiranju obeh komunikacijskih protokolov.

7.2.1 Skladi

Največji problem pri vgradnji samih protokolnih modulov je problem zaradi omejenosti s skladi (stack). Števci uporabljajo namreč operacijski sistem. MT830 in MT831 uporabljata operacijski sistem (OS) razvit v Iskraemeco in se imenuje JOS, MT860 pa ECOS, ki je brezplačni OS dosegljiv za vložene (embedded) sisteme. Vsak izmed operacijskih sistemov ima v svojem delovanju definirane naloge (task), katerim je dodeljen določen obseg sklada. Ker imamo v mikrokrmilniških sistemih omejena sredstva (resource) smo omejeni tudi s pomnilnikom, kar pa pomeni, da je tudi velikost sklada omejena. Ravno zaradi tega dejstva moramo pisati kodo, ki uporablja čim manj skladovnega pomnilnika.

7.2.2 Dostopanje do podatkov "Big endian - little endian"

Gre za problem pri dveh različnih načinih dostopanja do podatkov ("Big" in "Little" endian). Na virtualnem števcu, ki deluje na PC, deluje zapisovanje po sistemu "Little endian". Tu se podatki zapisujejo v pomnilnik v smeri najpomembnejšega (MSB) zloga (byte-a) do najmanj pomembnega (LSB se tako zapiše na najnižjo pomnilniško lokacijo). Pri "Big endian" načinu pa je smer ravno obratna. Ravno ta način pa se uporablja na števcih. Pri reševanju tega problema moramo biti zgolj pozorni za kakšen način dostopanja do podatkov gre in obrniti podatke po potrebi.

7.2.3 Programska koda

Vzrok problema se zopet nahaja v omejenosti sredstev mikrokrmilniških sistemov Zaradi tega je potrebno kodo pisati izredno optimalno in učinkovito. Programski jezik, ki omogoča pisanje optimalne kode je ANSI C programski jezik, zato je bil tudi uporabljen pri programiranju naših števcev. Komunikacijski protokoli so zelo kompleksni, zato posledično uporabljajo največ delovnega spomina, najbolje bremenijo sklad in "producirajo" tudi veliko programske kode. Ravno zaradi omenjenih omejenosti je implementacija komunikacijskih protokolov IEC870-5-102 in DLMS/COSEM toliko bolj zahtevna.

7.2.4 Velikost medpomnilnika (buffer)

Velikot medpomnilnika (buffer) se določa glede na velikost podatkovnega okvirja, ki ga pošiljamo. Za večje podatkovne okvirje je potrebno alocirati večje podatkovne medpomnilnike. V obeh primerih (IEC870-5-102 in DLMS/COSEM protokol) je tipična velikost podatkovnega okvirja okoli 128 byte-ov. Iz tega sledi, da je potrebna velikost medpomnilnika enaka. Tu se pojavi problem, ker imamo zopet omejenost s strani mikrokrmilniških sistemov. Na voljo ni veliko delovnega pomnilnika, zato je velikost medpomnilnika omejena (tipično okoli 128 byte-ov). Potrebno je omeniti, da potrebuje protokol vsaj dva medpomnilnika. Enega za sprejemanje zahteve in drugega za oddajanje odgovora. Seveda je možna tudi druga pot, ko imamo na razpolago le en medpomnilnik, je pa zaradi tega koda toliko bolj kompleksna.

7.3 Testiranje

Pomembno je poudariti, da je testiranje programske kode izredno pomembno. Da program lahko vstopi v neko ciljno napravo ali na trg je potrebno veliko testiranja, verificiranja in to na več nivojih.

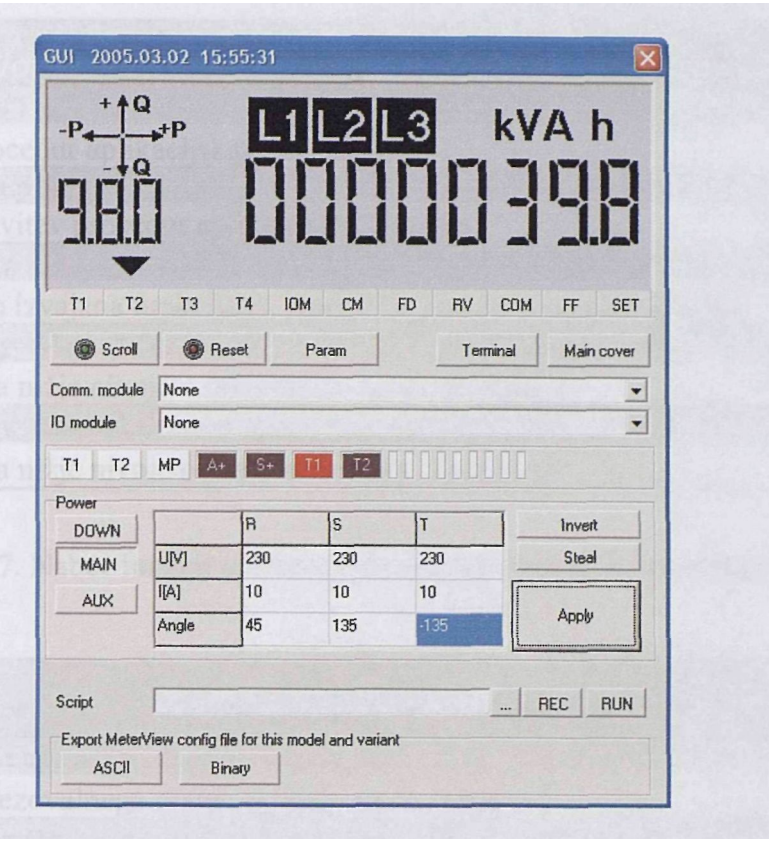
Testiranje programske kode v Iskraemeco poteka na več nivojih. Najprej testira programsko kodo sam razvojni inženir z ustreznimi testnimi aplikacijami. Programska koda je že vgrajena v števec (ciljna naprava) tako, da se s tem preverja odziv celotne naprave na novovgrajeno kodo. Ko se taka koda preveri na tem nivoju, testira števec druga razvojna skupina. Na tem mestu se števec konfigurira po specifikacijah stranke in že predstavlja več ali manj zaključen izdelek. Sledi testiranje pri neodvisni skupini testnega laboratorija, kjer se testira ali števec ustreza internim standardom (verifikacija). Interni standardi so vedno strožji od kriterijev, ki

jih določajo stranke ali splošno sprejeti normativi. Ko tako preverjen števec zapusti podjetje se še zadnjič preverja pri stranki.

Poleg vseh naštetih preverjanj, pa lahko obstajajo še dodatne testne procedure, ki se jih poda s specifikacijo komunikacijskega protokola (tak primer je DLMS/COSEM).

7.3.1 Nivoji testiranja

Testiranje programskega modula je potekalo na dveh nivojih. Sprva je bila koda testirana na virtualnem števcu. Gre za aplikacijo, ki je po funkcionalnosti enaka realnemu števcu s specifičnimi razlikami. Na nivoju aplikativnega in modularnega dela se realni in virtualni števec ne razlikujeta. Na nivoju platforme pa se razlikujeta v tem, da virtualni števec uporablja WinXP OS, realni števec pa JOS. Razlika je torej v datotečnem sistemu, medijih shranjevanja podatkov (realni števec uporablja FLASH, FRAM), podatkovnih tipih, OS-u itd. Razlika je zelo pomembna. WinXP OS, s stališča vloženih sistemov, skorajda nima omejitve sredstev. Neomejena velikost RAM-a, skladov, hitrosti procesorja itd. To je dejstvo, ki je razvoj obeh komunikacijskih modulov močno omejevalo. Praktično to pomeni, da koda protokolov deluje na osebnem računalniku brez zapletov, ko pa jo prenesemo na ciljno napravo (tu so sredstva močno omejena) pa ne deluje nič več. Slika 36 prikazuje industrijski elektronski virtualni števec električne energije.



Slika 36. Elektronski virtualni števec električne energije.

7.4 Umestitev IEC870-5-102 komunikacijskega protokola v delovno okolje

7.4.1 Vgradnja protokola v števec

Podobno kot je bila predstavljena zgradba protokola po OSI modelu (3 nivoji v primeru IEC870-5-102), je tudi zgrajen programski algoritem protokola. Programski algoritmi so bili programirani v ANSI C programskem jeziku in implementirani na elektronskem trifaznem industrijskem električnem števcu MT830, MT831 in MT860.

7.4.2 Struktura programske kode po OSI modelu

Kot je bilo že omenjeno sledi zgradba protokola OSI modelu. Izvorna koda je bila napisana prav tako v treh nivojih. Posamezen nivo predstavlja zaključeno celoto, ki pa lahko dostopa do ostalih nivojev po urejeni hierarhiji. Vsak nivo ima nabor C funkcij, ki definirajo delovanje nivoja. Nabor funkcij in struktura datotek po treh nivojih prikazujejo tabele 7, 8 in 9. Funkcije ne vsebujejo uporabljenih argumentov zaradi boljše razumljivosti.

1. Aplikacijski nivo:	srv870app.c, srv870app.h
`Srv870_App_Init();`	
inicializacija procedur aplikacijskega nivoja	
`Srv870_App_Cleanup();`	
dealociranje, ustavitev procedur aplikacijskega nivoja	
`Srv870_App_Run();`	
glavna periodično izvajana funkcija za zajemanje zahtev kontrolne postaje	
`Srv870_App_Notify_Class1();`	
funkcija informira nižje nivoje o prisotnosti class 1 podatkov	
`Srv870_App_Notify_Class2();`	
funkcija informira nižje nivoje o prisotnosti class 2 podatkov	

Tabela 7. Nabor funkcij aplikacijskega nivoja in datotečna struktura.

2. Povezovalni nivo:	iec870_5_102.c, iec870_5_102.h, iec870_5_102_par.h, iec870_5_102_cfg.h
`IEC870_5_102_Start();`	
inicializacija povezovalnega nivoja (vezana na fizični nivo)	
`IEC870_5_102_Stop();`	
dealociranje, ustavitev procedur povezovalnega nivoja (vezano na fizični nivo)	
`IEC870_5_102_GetRequest();`	
glavna periodično izvajana funkcija za zajemanje prihajajočih zahtev iz fizičnega nivoja	
`IEC870_5_102_SetResponse();`	
funkcija, ki posreduje odgovor aplikacijskega nivoja fizičnemu nivoju	
`IEC870_5_102_ASDU_Init();`	
funkcija postavi ustrezne podatke v glavi (head) ASDU podatkovne enote	
`IEC870_5_102_ASDU_Append();`	
funkcija doda podatke ASDU podatkovni enoti	

Tabela 8. Nabor funkcij povezovalnega nivoja in datotečna struktura.

3. Fizični nivo:	iec870_5.c, iec870_5.h, iec870_5_par.h, iec870_5_cfg.h
`IEC870_5_Start();`	
inicializacija fizičnega nivoja (inicializacija serijskega vhodno/izhodnega vmesnika oziroma UART-a - Universal Asynchronous Receiver-Transmitter)	
`IEC870_5_Stop();`	
dealociranje, ustavitev procedur fizičnega nivoja (sprostitev UART-a)	
`IEC870_5_GetRequest();`	
glavna periodično izvajana funkcija za zajemanje prihajajočih zahtev s strani kontrolne postaje (SCADA)	
`IEC870_5_SetResponse();`	
funkcija posreduje odgovor povezovalnega nivoja SCADA-i	

Tabela 9. Nabor funkcij fizičnega nivoja in datotečna struktura.

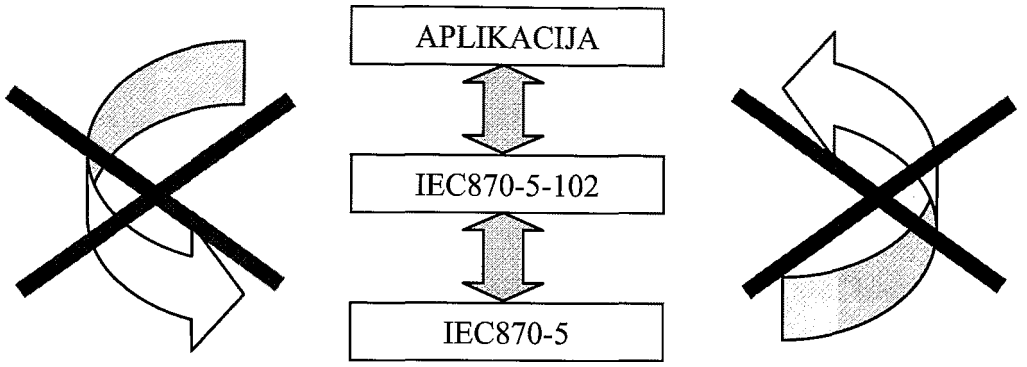
V primera IEC870-5-102 komunikacijskega protokola ni bilo večjih problemov z vgrajevanjem modula v števec. Sam protokol ni tako potraten kot je DLMS/COSEM, zato je bilo potrebno le malo povečati sklad. V obdobju testiranja števca z vgrajenim IEC870-5-102 protokolom, je bilo potrebno popraviti le še "big, little endian" problem pri dostopanju do podatkov. Problem je bil rešen z ustreznim popravkom kode.

V nadaljevanju bo opisana implementacija aplikacije, oddajne in sprejemne procedure protokola.

7.4.3 Delovanje aplikacije IEC870-5-102

Aplikacija je na nivoju kontroliranega sistema (števec) pravzaprav algoritem, ki skrbi za pravilno delovanje celotnega IEC870 protokola. Aplikacija tako komunicira z IEC870-5-102 protokolom (povezovalni nivo), ta pa naprej z IEC870-5 (fizični nivo) protokolom. V tem primera je IEC870-5-102 protokol namenjen olajšanju delovanja aplikacije, saj ji ne dopušča (v programskem smislu) dostopa in vpogleda v IEC870-5 programski modul. Na ta način se doseže poenostavljena uporaba protokola. Protokol je na ta način tudi veliko bolj univerzalen ia fleksibilen, deluje namreč kot vmesnik (API) in ga je možno prilagoditi ali zamenjati za katerikoli način izvedbe aplikativnega nivoja (aplikacije). Preprosto to pomeni, da lahko modul zamenjamo z drugim, če tako zahteva spremenjen aplikativni nivo. Fizični nivo tako ostane popolnoma nespremenjen.

Oba dela (povezovalni in fizični nivo) služita čim lažjemu prenosu podatkov od kontrolne do kontrolirane postaje (sistema). Slika 37 prikazuje medsebojno odvisnost aplikativnega nivoja protokola ter obeh IEC870 protokolov.



Slika 37. Medsebojna odvisnost aplikacije, IEC870-5 in IEC870-5-102 protokola.

7.4.4 Delovanje sprejemne in oddajne procedure

Sprejemna procedura, ki je natančneje obrazložena v poglavju 5.4.1.1, skrbi za sprejem vseh podatkov poslanih po podatkovnem omrežju. Poskrbi tudi za identifikacijo samega naslova podatkovnega okvirja, z drugimi besedami, ugotovi ali je podatkovni okvir namenjen napravi, kjer se sprejemna procedura nahaja.

Sprejemna procedura mora poskrbeti tudi za nekatere samodejne odzive za katere ni nujno, da aplikativni nivo intervenira. Tako je dodeljena večja avtonomnost programskemu modulu IEC870, pa tudi aplikacija je manj obremenjena.

Oddajna procedura je tako deloma podrejena sprejemni proceduri, drugače pa jo aktivira aplikacija. Seveda aplikativni nivo aktivira sprejemno proceduro le v primerih, da to od nje zahteva ali prosi kontrolna postaja (SCADA), v nasprotnem primeru se v podatkovno omrežje ne sme pošiljati nobenih podatkov.

7.4.5 Testne aplikacije in procedure

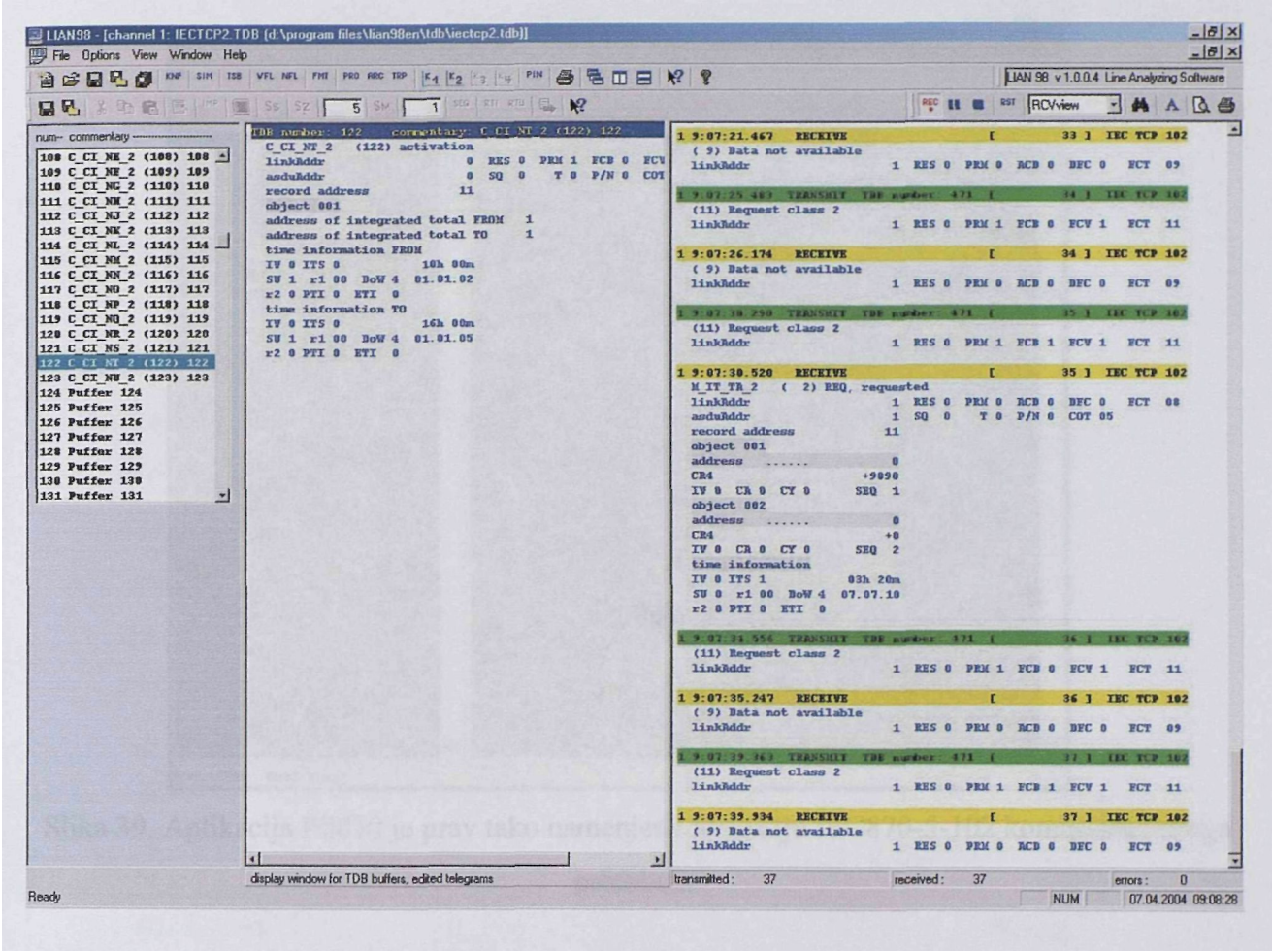
Testiranje komunikacijskega modul IEC870-5 in IEC870-5-102 protokola je potekalo s pomočjo aplikacije LIAN98 in PS870. Programska koda za oba protokola je bila testirana zgolj na nivoju razvojnega inženirja in razvojne skupine.

7.4.5.1 LIAN 98

S pomočjo testne aplikacije LIAN (Slika 38), je bil testiran IEC870-5-102 komunikacijski protokol. Testna aplikacija LIAN omogoča testiranje vseh ASDU podatkovnih okvirjev, ki so definirani v IEC870-5-102 protokolu, in tudi tistih, ki si jih uporabnik protokola poljubno definira. Tako orodje je sicer komplicirano za uporabo a je tudi izredno učinkovito in podrobno. Orodje omogoča tudi generiranje poljubnih sekvenc. To pomeni, da lahko za poljuben čas definiramo katere ASDU podatkovne enote naj LIAN pošilja. Pomembna lastnost orodja je simuliranje kontrolirnega sistema oziroma SCADA-e.

Na skrajnem levem delu vmesniškega okna aplikacije (GUI - Graphical User Interface; Slika 38) izberemo ASDU, na sredinskem delu se izpišejo vse pomembne lastnosti podatkovne enote, na tretjem, skrajno desnem delu okna, pa se izpisujejo oddani in prejeti podatkovni okvirji. Vsak okvir, ki ga LIAN98 sprejme ali odda vsebuje informacije o glavi ASDU podatkovne enote (tip, sekvenca, vzrok, naslov, naslov zapisa, postavljeni statusi itd.), vsebovane podatke in informacijo o morebitnih napakah.

Edina omejitev, ki jo ima LIAN98, je ta, da je orodje testna različica ("demo") . Namreč, orodje dovoljuje sprejemanje in pošiljanje le največ 64 podatkovnih okvirjev.

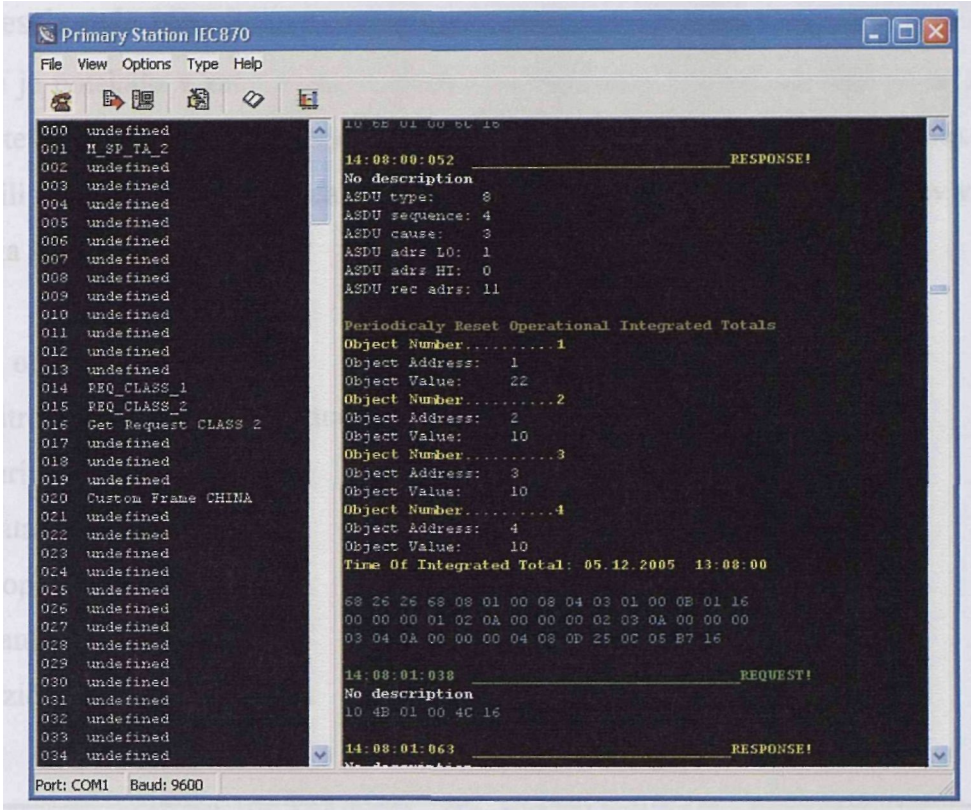


Slika 38. Aplikacija LIAN namenjena testiranju IEC870-5-102 komunikacijskega protokola.

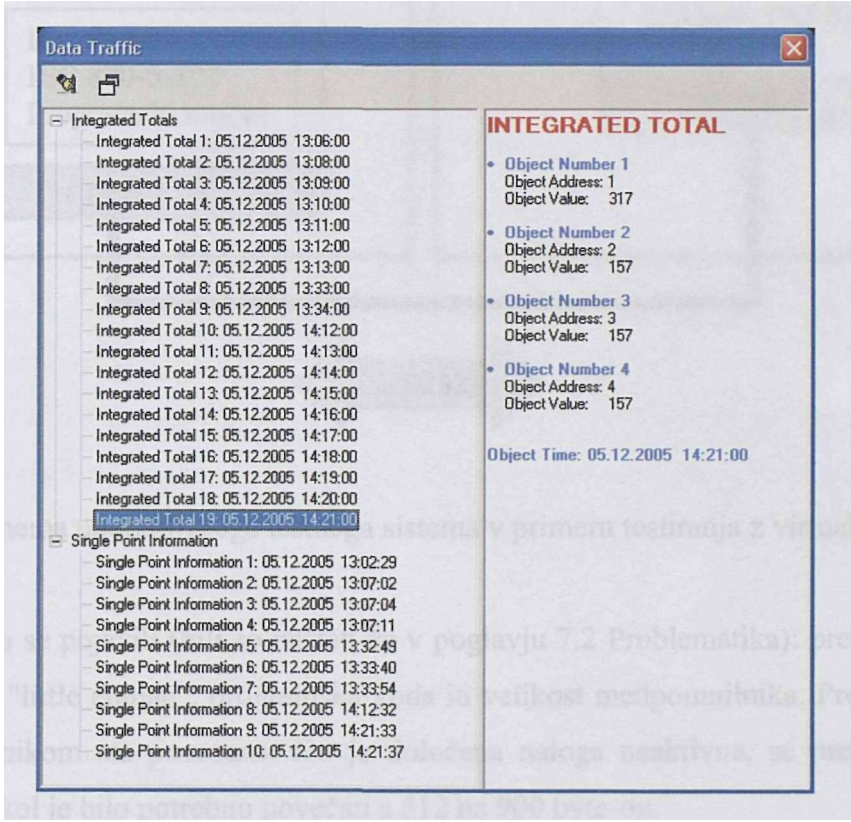
7.4.5.2 Primary station 870 - PS870

PS870 aplikacija (Slika 39) je po funkcionalnosti zelo podobna LIAN aplikaciji. Ker je uporabljeni LIAN zgolj predstavitvena ("demo") verzija, se je za dolgotrajnejše testiranje uporabljal PS870. PS870 tako omogoča časovno gledano dolgotrajno testiranje. Omogoča tudi sledenje poslanim podatkovnim okvirjem, pregledovanje njihovi vsebini, analiziranje in generiranje lastnih okvirjev. Nosi vse informacije, ki so bile omenjene že popreje pri LIAN98 aplikaciji.

PS870 aplikacija je bila razvita v okviru magistrske naloge oziroma v okviru projekta integracije komunikacijskega protokola IEC870-5-102 v elektronski števec.



Slika 39. Aplikacija PS870 je prav tako namenjena testiranju IEC870-5-102 komunikacijskega protokola.



Slika 40. Del aplikacije PS870, ki prikazuje podatkovni promet na liniji.

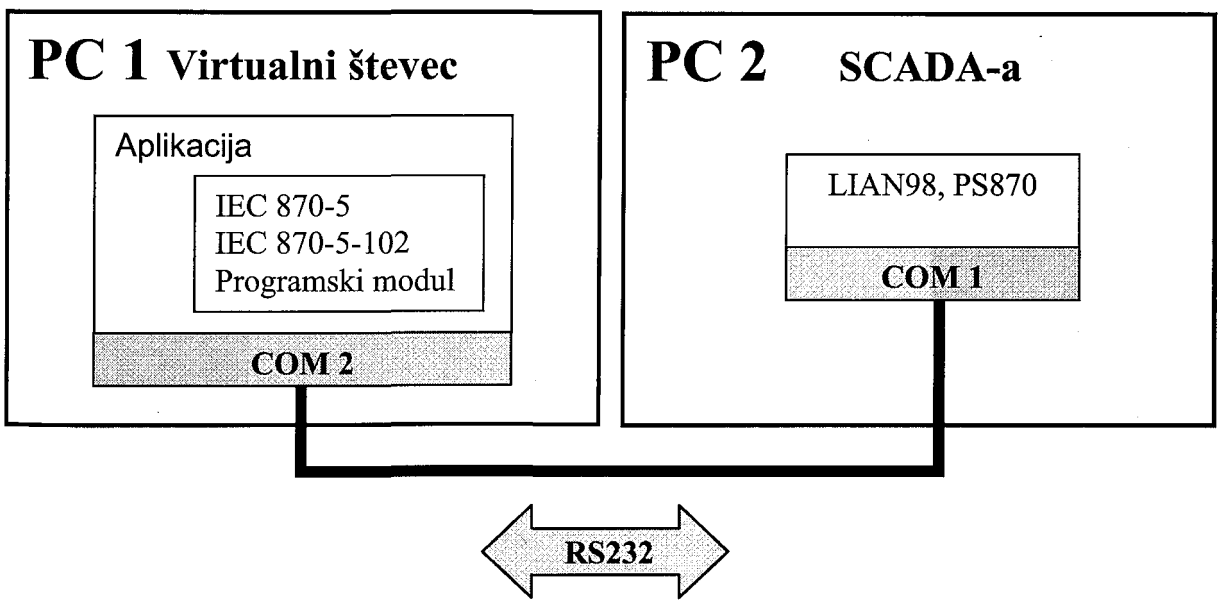
7.4.6 Testiranje IEC 870-5-102 programskega modula

Testiranje je potekalo v začetni fazi s pomočjo virtualnega števca, pozneje pa še na samem realnem števcu. Kot je bilo že omenjeno večjih težav na virtualnem števcu ni bilo. Problemi so nastopili pri prenosu kode na realni števec. Sliki 41 in 42 prikazujeta postavitev virtualnega in realnega števca pri testiranju.

Pogoji za oba testiranja:

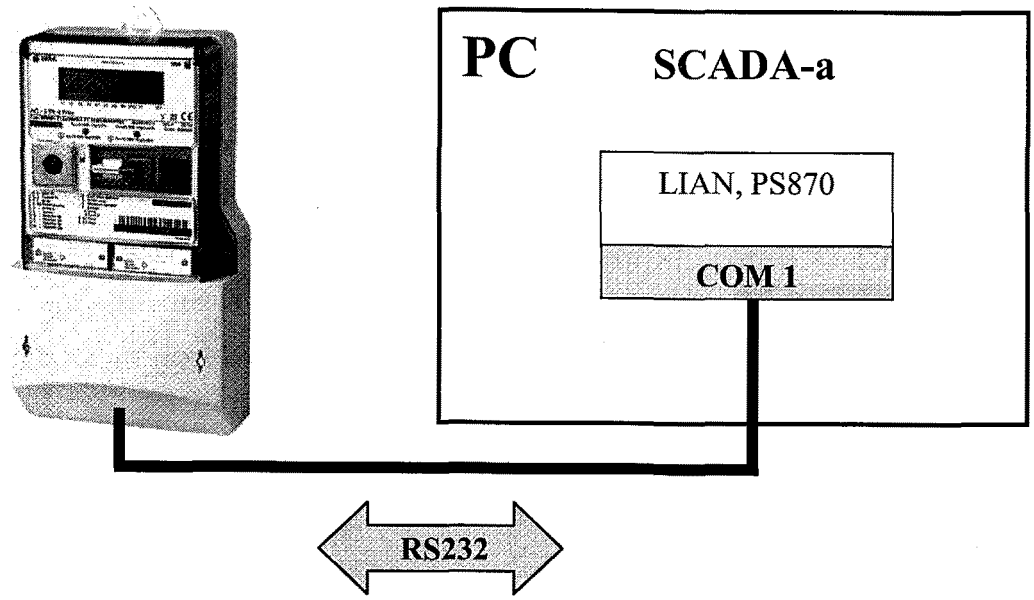
- Hitrost povezave: 9600 Baud
- Pariteta: NONE
- Data bits: 8
- Stop bit: 1.5
- Handshake: NONE

  Fizični medij: RS232



Slika 41. Shema uporabljenega testnega sistema v primeru testiranja z virtualnim števcem.

Problemi, ki so se pojavili (bili so našteti že v poglavju 7.2 Problematika): premajhni skladi, "big endian" - "little endian", programska koda in velikost medpomnilnika. Problem s skladi in medpomnilnikom sta povezana. Ko je določena naloga neaktivna, se preloži na sklad. Sklad za protokol je bilo potrebno povečati s 512 na 900 byte-ov.



Slika 42. Shema uporabljenega testnega sistema v primeru testiranja z realnim števcem.

Komunikacija poteka preko fizičnega vmesnika (RS232, RS485, tokovna zanka itd.) in z IEC870-5 komunikacijskim protokolom.

7.4.7 Testni rezultati

Komunikacijski modul je bil implementiran na ciljni napravi - industrijski elektronski števec električne energije (MT830, MT831, MT860). S pomočjo omenjenih aplikacij je bil simuliran SCADA sistem, na katerega je bil priklopljen en števec.

Delovanje števca:

- števec se inicializira ustrezno po protokolu
- števec pošilja CLASS 2 podatke (ti na števcu ustrezajo podatkom bremenske krivulje (Load Profile): čas zajetega zapisa, podatki o merjenih veličinah (energija), dogodki)
- števec pošilja CLASS 1 podatke: to so različni dogodki, ki se zgodijo ob delovanju števca (testiranje bil obračun, odprtje/zaprtje glavnega pokrova in pokrova priključnice, izklop in vklop števca)

Potek komunikacije (odgovori na zahteve) je razviden iz prilog:

- Inicializacija komunikacije: Priloga 9.1
- Zajem podatkov razreda 2 (Class 2 data): Priloga 9.2
- Zajem podatkov razreda 1 (Class 1 data): Priloga 9.3

7.5 Umestitev DLMS/COSEM protokola v delovno okolje

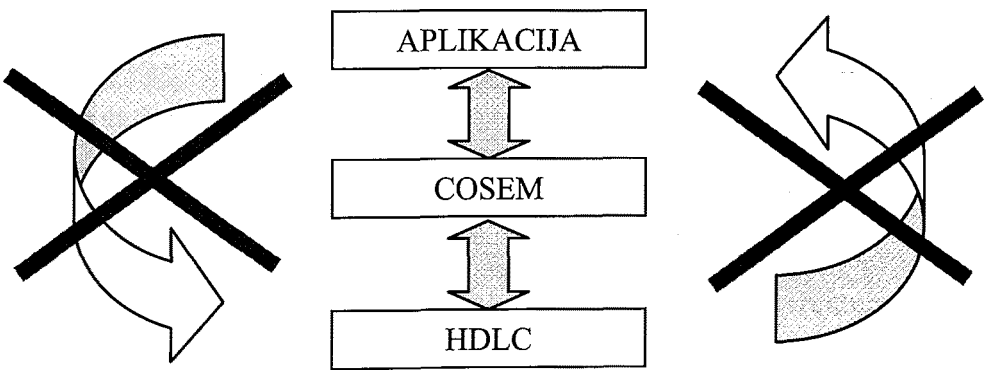
Že branje same DLMS/COSEM specifikacije nakazuje obseg in kompleksnost protokola. Predvideva namreč vse možne situacije pri izmenjavi podatkov in je napisan univerzalno. Ravno zaradi tega dejstva je koda protokola bolj obsežna in kompleksnejša od IEC870-5-102. Posledično porabi več mikrokrmilniških sredstev. Konkretno to pomeni (v primerjavi z IEC870-5-102) krepko povečan sklad in obsežnejšo programsko kodo. Med testiranjem kode na števcu se je pokazal isti problem kot pri vgradnji IEC870-5-102 protokola, to je dostopanje do podatkov.

7.5.1 Vgradnja protokola v števec

Tudi komunikacijski del DLMS/COSEM protokola sloni na OSI modelu (le 3 nivoji kot je v primera IEC870-5-102) in tako je tudi zgrajen programski algoritem protokola (programski algoritmi so bili programirani v ANSI C programskem jeziku). Koda je bila implementirana na elektronskem trifaznem industrijskem električnem števcu MT830 in MT831.

7.5.2 Struktura programske kode po OSI modelu

Zgradba protokola je enaka zgradbi IEC870-5-102 protokola. Se pravi, da posamezen nivo predstavlja zaključeno celoto in lahko dostopa do ostalih nivojev po urejeni hierarhiji (slika 43).



Slika 43. Medsebojna odvisnost aplikacije, IEC870-5 in IEC870-5-102 protokola.

Prav tako ima vsak nivo nabor ustreznih C funkcij, ki definirajo delovanje nivoja. Nabor funkcij in struktura datotek po treh nivojih prikazujejo tabele 10, 11 in 12. Funkcije ne vsebujejo uporabljenih argumentov zaradi boljše razumljivosti.

1. Aplikacijski nivo:	SrvDLMSApp.c, SrvDLMSApp.c
`SrvDLMS_App_Init();`	
dinamična inicializacija procedur aplikacijskega nivoja	
`SrvDLMS_App_Init_Static();`	
statična inicializacija procedur aplikacijskega nivoja	
`SrvDLMS_App_Cleanup();`	
dinamično dealociranje, ustavitev procedur aplikacijskega nivoja	
`SrvDLMS_App_Cleanup_Static();`	
statično dealociranje, ustavitev procedur aplikacijskega nivoja	
`SrvDLMS_App_Run();`	
glavna periodično izvajana funkcija za zajemanje zahtev	

Tabela 10. Nabor funkcij aplikacijskega nivoja in datotečna struktura.

2. Povezovalni nivo	cosem.c, cosem.h, cosem_par.h, cosem_cfg.h
`COSEM_start();`	
dinamična inicializacija procedur povezovalnega nivoja (vezana na fizični nivo)	
`COSEM_start_static();`	
statična inicializacija procedur povezovalnega nivoja (vezana na fizični nivo)	
`COSEM_stop();`	
dinamično dealociranje, ustavitev procedur povezovalnega nivoja (vezano na fizični nivo)	
`COSEM_stop_static();`	
statično dealociranje, ustavitev procedur povezovalnega nivoja (vezano na fizični nivo)	
`COSEM_get_request();`	
periodično izvajana funkcija za zajemanje zahtev (vezana na fizični nivo)	
`COSEM_set_response();`	
funkcija za generiranje odgovor na ustrezno zahtevo (vezana na fizični nivo)	
`COSEM_pdu_decode();`	
dekodiranje podatkov vhodne APDU podatkovne enote (COSEM mapper)	
`COSEM_pdu_encode();`	
kodiranje podatkov izhodne APDU podatkovne enote (COSEM mapper)	

Tabela 11. Nabor funkcij povezovalnega nivoja in datotečna struktura.

3. Fizični nivo:	hdlc.c, hdlc.h, hdlc_par.h, hdlc_cfg.h
`HDLC_Open();`	
dinamična inicializacija fizičnega nivoja (UART itd.)	
`HDLC_Open_Static();`	
statična inicializacija fizičnega nivoja (UART itd.)	
`HDLC_Close();`	
dinamično dealociranje, ustavitev procedur fizičnega nivoja (sprostitev UART-a)	
`HDLC_Close_Static();`	
statično dealociranje, ustavitev procedur fizičnega nivoja (sprostitev UART-a)	
`HDLC_GetRequest();`	
glavna periodično izvajana funkcija za zajemanje prihajajočih zahtev	
`HDLC_SetResponse();`	
funkcija za generiranje odgovor na ustrezno zahtevo	

Tabela 12. Nabor funkcij fizičnega nivoja in datotečna struktura.

V naboru funkcij obstajata dva tipa funkcij. Prvi je dinamični drugi pa statični tip. Dinamični nabor funkcij se uporablja za dinamično alociranje protokolnih sredstev in uporablja svoj sklad. V primera statičnega alociranja pa DLMS/COSEM uporablja sklad IEC62056-21 protokola, ki je tudi vgrajen v števec. Slednja možnost velja le, ko sta oba protokola prisotna v števcu. Na ta način se varčuje s skladi.

7.5.3 Testne aplikacije in procedure

Testiranje komunikacijskega modula DLMS/COSEM je potekalo z aplikacijo Meter View.

7.5.3.1 Programski Paket METERVIEW

Osnovni namen programskih orodij za parametriranje elektronskih števcev je nastavljanje parametrov in preizkušanje na čimbolj pregleden in enostaven način. Ta orodja so običajno namenjena elektro-distribucijam za uporabo v njihovih umerjevalnicah. Uporabniku prijazen način nastavljanja parametrov zahteva naslednje:

- avtomatsko zaznavanje tipa števca, da lahko program brez posredovanja uporabnika odpre podatke o števcu (projekt), od katerih je odvisen tudi nabor parametrov
- pregleden prikaz parametrov na obrazcu z možnostjo popravljanja
- vpisovanje parametrov v števec posamično, po skupinah ali celotni nabor

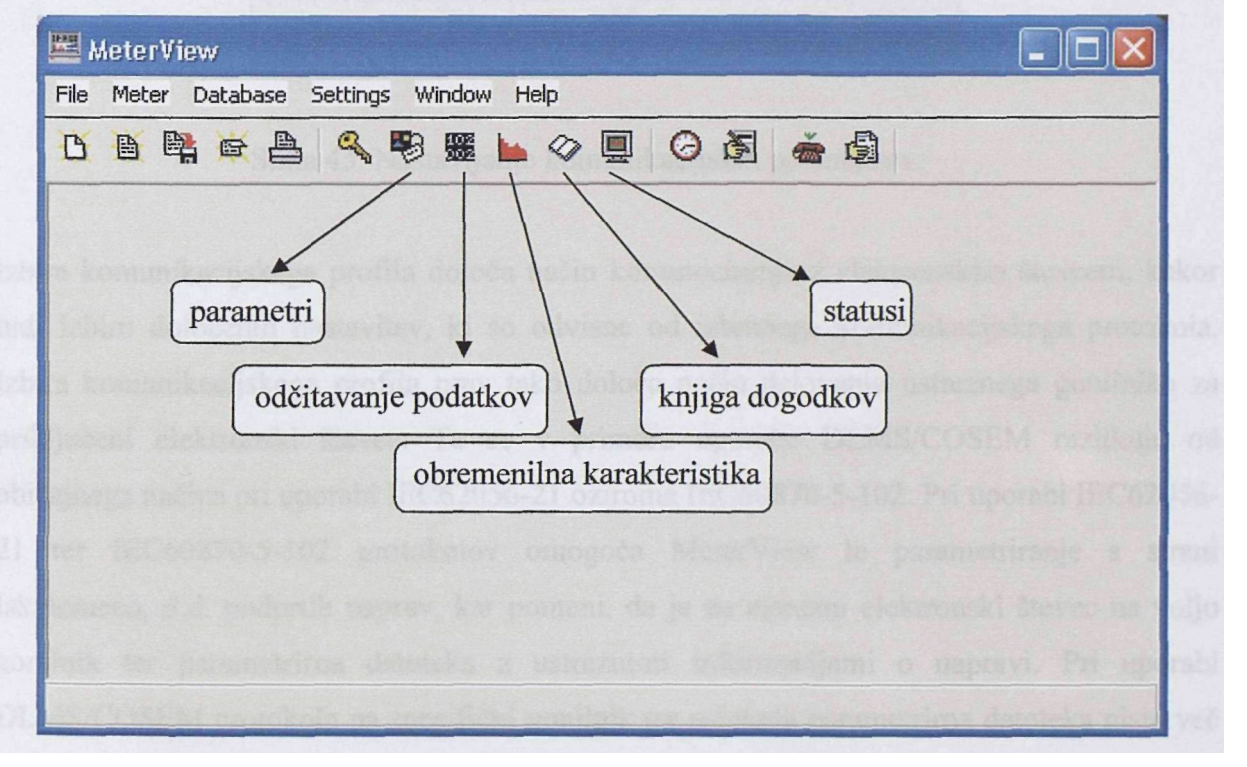
Ravno za te namene se je v Iskraemeco d.d. razvil programski paket Meter View, ki ima omenjene lastnosti. Testiranje komunikacijskega modula je tako potekalo s programskim

paketom Meter View 4.0.7. Meter View je načrtovan za poenostavitev nastavljanja parametrov in preizkušanje elektronskih števcev električne energije, ki jih izdeluje Iskraemeco, d.d. Glavne značilnosti paketa:

- parametriranje različnih tipov števcev preko enotnega uporabniškega vmesnika
- različna nadzorna odčitavanja za potrebe preizkušanja števcev
- podpora različnim komunikacijskim protokolom (IEC62056-21, IEC870-5-102 Spain, DLMS/COSEM) in medijem (RS232, CS, optika, modem)

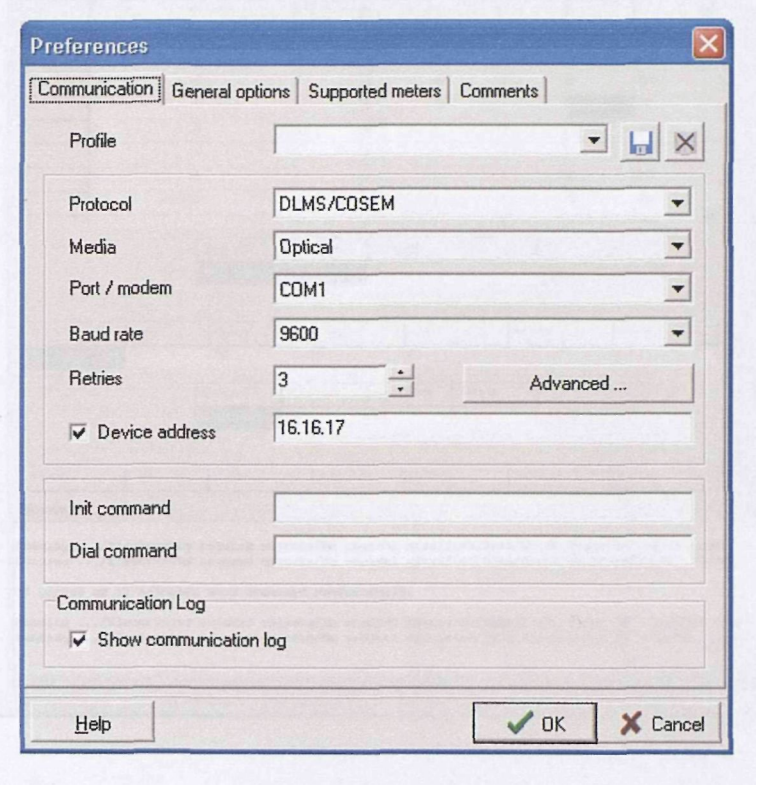
Uporabniški vmesnik programa vsebuje glavno MDI okno (slika 44), v okviru katerega se lahko izvaja:

- dostopanje do parametrov (Parameters)
- odčitavanje podatkov (DRO - Data Read Out)
- zajemanje obremenilne karakteristike (Load profile)
- zajemanje knjige dogodkov (Log book)
- dostopanje do statusov (Status)



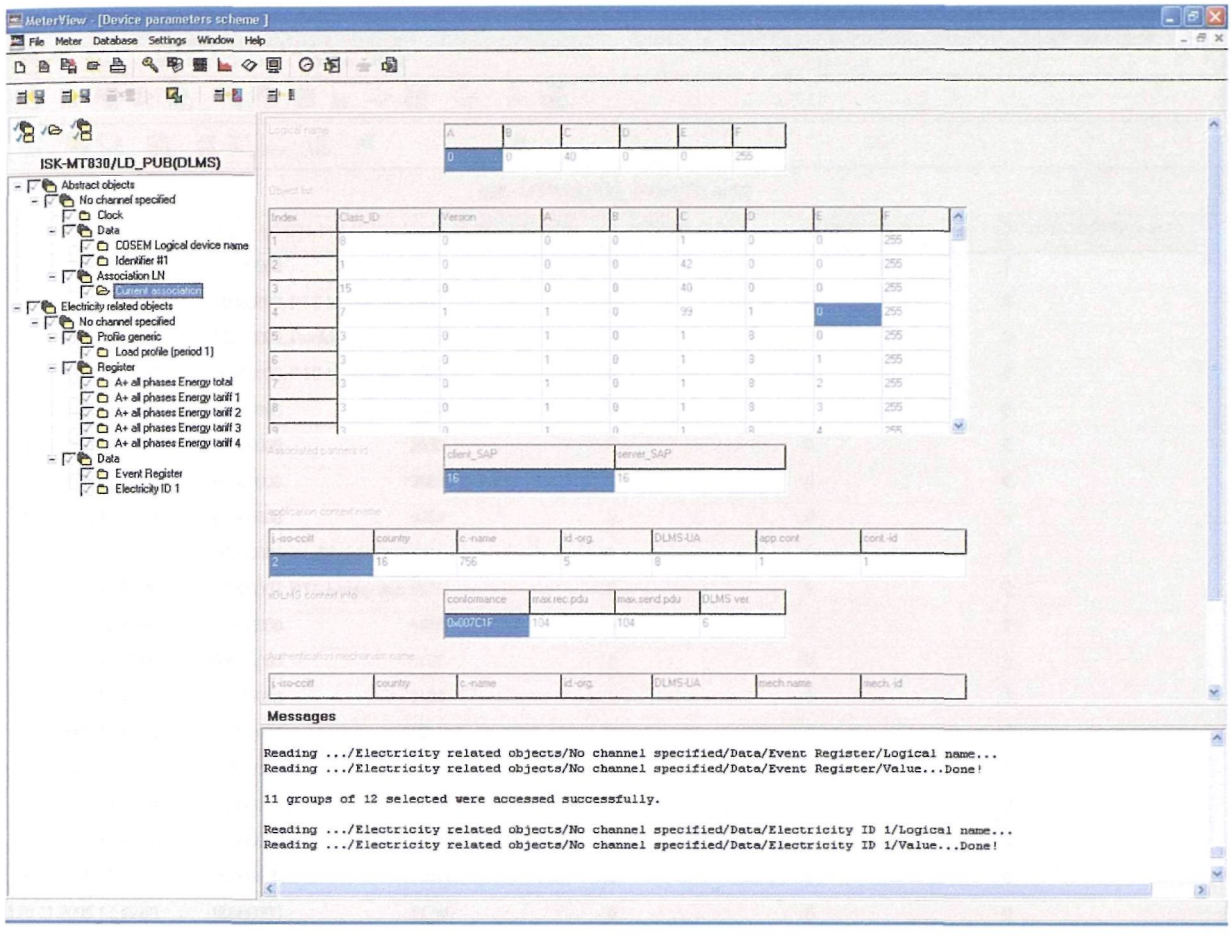
Slika 44. Glavno okno, meni in orodna vrstica

Nastavitev želenega komunikacijskega profila se opravi preko nastavitvenega okna. Nastavitveno okno nudi uporabniku možnost nastavitev komunikacijskih parametrov, kot prikazuje slika 45.

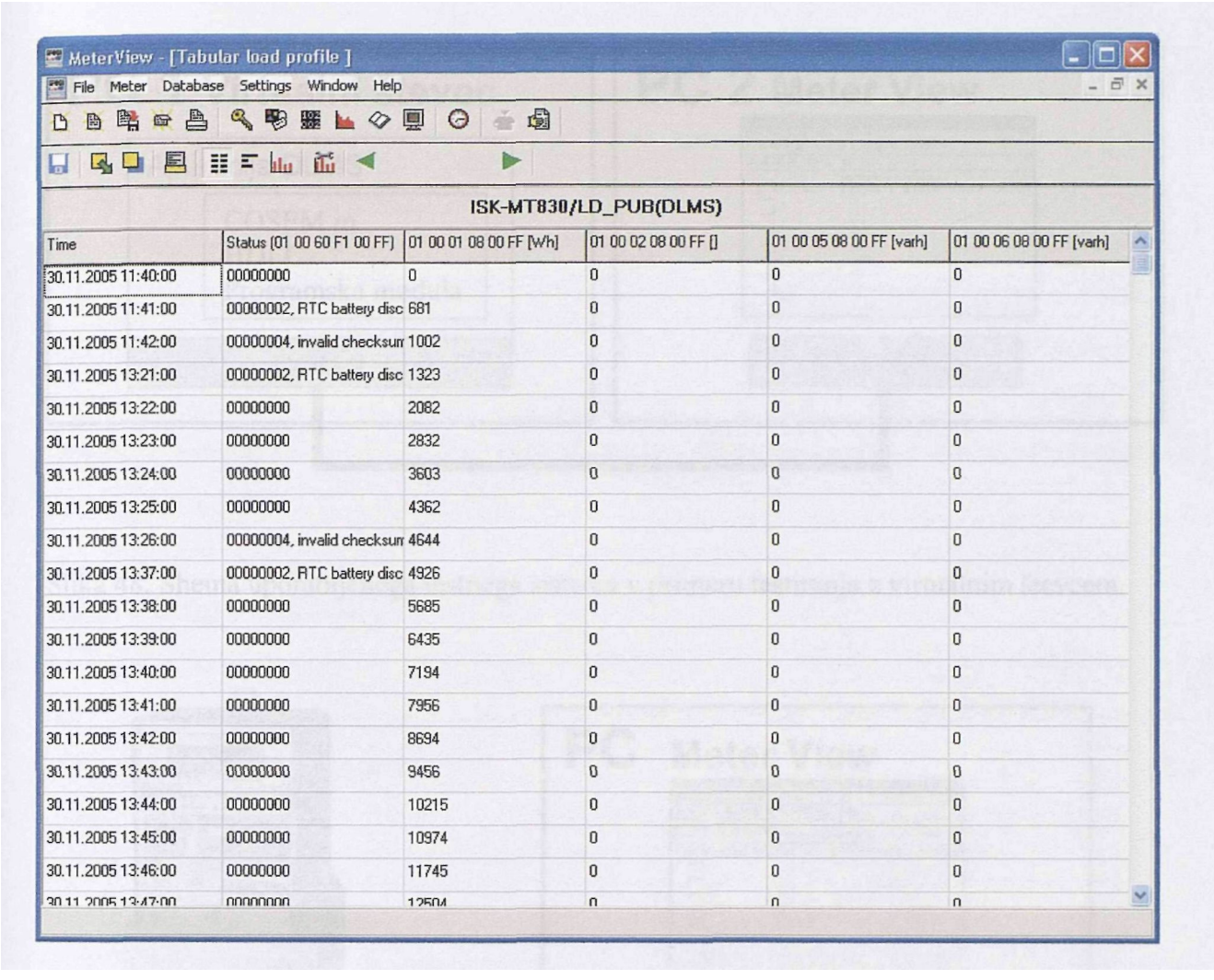


Slika 45. Nastavljanje komunikacijskih parametrov.

Izbira komunikacijskega profila določa način komuniciranja z elektronskim števcem, kakor tudi izbiro določenih nastavitev, ki so odvisne od izbranega komunikacijskega protokola. Izbira komunikacijskega profila prav tako določa način delovanja ustreznega gonilnika za priključeni elektronski števec. Ta se v primeru uporabe DLMS/COSEM razlikuje od običajnega načina pri uporabi IEC62056-21 oziroma IEC60870-5-102. Pri uporabi IEC62056-21 ter IEC60870-5-102 protokolov omogoča MeterView le parametriranje s strani Iskraemeco, d.d. podprtih naprav, kar pomeni, da je za ustrezni elektronski števec na voljo gonilnik ter parametrirna datoteka z ustreznimi informacijami o napravi. Pri uporabi DLMS/COSEM protokola pa specifični gonilnik ter ustrezna parametrirna datoteka nista več potrebni, saj DLMS naprava nudi vso informacijo za sestavo modela števca v generičnem gonilniku. Zajemanje podprtih registrov za primer naprave, ki podpira DLMS/COSEM protokol, prikazuje slika 46, slika 47 pa zajete podatke o bremenski krivulji (load profile).



Slika 46. Okno za zajemanje podprtih registrov.



Time	Status (01 00 60 F1 00 FF)	01 00 01 08 00 FF [Wh]	01 00 02 08 00 FF []	01 00 05 08 00 FF [varh]	01 00 06 08 00 FF [varh]
30.11.2005 11:40:00	00000000	0	0	0	0
30.11.2005 11:41:00	00000002, RTC battery disc	681	0	0	0
30.11.2005 11:42:00	00000004, invalid checksum	1002	0	0	0
30.11.2005 13:21:00	00000002, RTC battery disc	1323	0	0	0
30.11.2005 13:22:00	00000000	2082	0	0	0
30.11.2005 13:23:00	00000000	2832	0	0	0
30.11.2005 13:24:00	00000000	3603	0	0	0
30.11.2005 13:25:00	00000000	4362	0	0	0
30.11.2005 13:26:00	00000004, invalid checksum	4644	0	0	0
30.11.2005 13:37:00	00000002, RTC battery disc	4926	0	0	0
30.11.2005 13:38:00	00000000	5685	0	0	0
30.11.2005 13:39:00	00000000	6435	0	0	0
30.11.2005 13:40:00	00000000	7194	0	0	0
30.11.2005 13:41:00	00000000	7956	0	0	0
30.11.2005 13:42:00	00000000	8694	0	0	0
30.11.2005 13:43:00	00000000	9456	0	0	0
30.11.2005 13:44:00	00000000	10215	0	0	0
30.11.2005 13:45:00	00000000	10974	0	0	0
30.11.2005 13:46:00	00000000	11745	0	0	0
30.11.2005 13:47:00	00000000	12504	0	0	0

Slika 47. Okno s prebranimi vrednostmi bremenske krivulje.

7.5.4 Testiranje DLMS/COSEM programskega modula

Razvoj kode in prvo testiranje je potekalo na virtualnem števcu, tako kot pri IEC870-5-102 protokolu. Pozneje je sledilo testiranje na samem realnem števcu. Sliki 48 in 49 prikazujeta postavitev virtualnega in realnega števca pri testiranju.

Pogoji:

Hitrost povezave: 9600 Baud

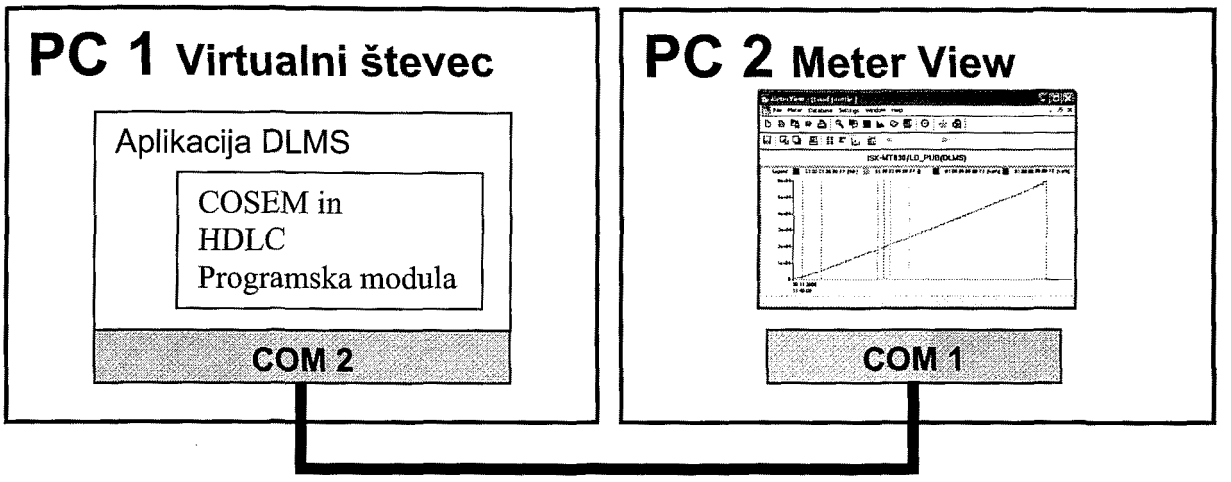
Pariteta: NONE

Data: 8

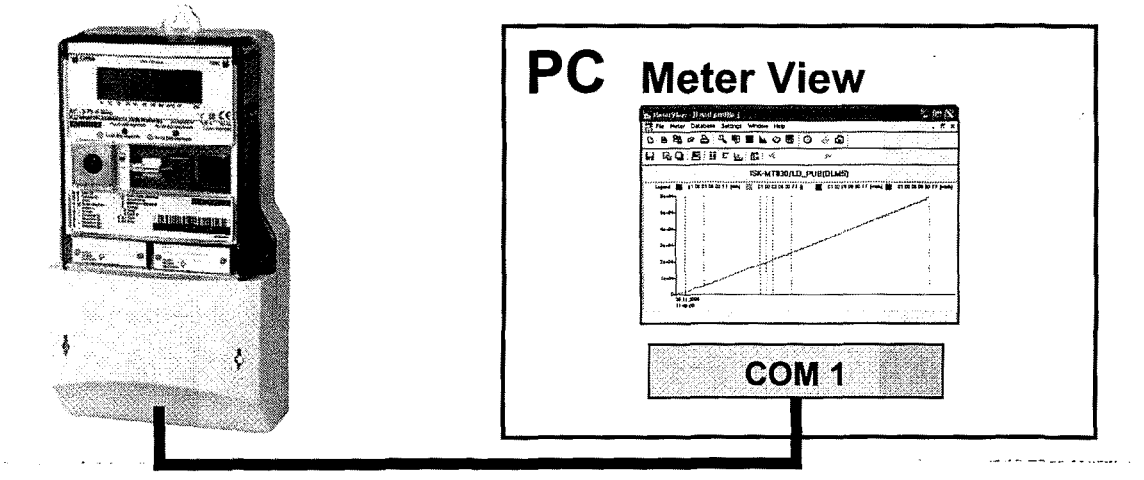
Stop bit: 1.5

Handshake: NONE

Fizični medij: RS232



Slika 48. Shema uporabljenega testnega sistema v primeru testiranja z virtualnim števcem.



Slika 49. Shema uporabljenega testnega sistema v primeru testiranja z realnim števcem.

Pri realnem števcu je bilo potrebno velikost sklada povečati na 1200 byte-ov. Za OS števca je to težka obremenitev, saj ima na razpolago le 8 kbyte-ov RAM-a. Povečanje sklada je vplivalo na stabilnost celotnega števca. Slednji problem se je odražal na stabilnosti komunikacijskega protokola IEC62056-21. Podobni problemi kot pri IEC870-5-102 protokolu, so sledili tudi tu. Problem zaradi različnega dostopanja do podatkov ("Big/Little endian") ni povzročal večjih težav saj je bil odpravljen le z manjšim popravkom kode. Večji problem je predstavljala obsežnost kode, predvsem pa velikost medpomnilnika.

Medpomnilnika sta bila dva (oddajni, sprejemni), to pa še ni vse. DLMS/COSEM protokol (podrobneje je o tem napisano v poglavju 6) je princip delovanja naprave. To pomeni, da mora biti ureditev kode in njeno delovanje po protokolu. V primeru naših števcev, kjer je ureditev modularna, koda ni grajena po specifikaciji. Iz tega sledi, da potrebujemo nek vmesnik (mapper), ki poskrbi za to transformacijo. Preprosto povedano je to del kode, ki podatke organizira po specifikacijah protokola. Omenjena koda je doprinesla še dodatne medpomnilnike, ki so bremenili procesorska sredstva. Prav tako je potrebno podatke ustrezno pripraviti, da so primerni za pošiljanje po komunikacijskih vodih. Koda, ki je opravljala to nalogo, je prav tako zahtevala svoj delež sredstev, kar je na koncu doprineslo k močni obremenitvi sklada.

7.5.5 Testni rezultati

Komunikacijski modul je bil implementiran na ciljni napravi - industrijski elektronski števec električne energije (MT830, MT831). S pomočjo Meter View paketa je potekala komunikacija z enim števcem.

Delovanje števca:

- s števcem se vzpostavi asociacija (osnovna; brez preverjanja pristnosti)
  števec omogoča zajemanje podprtih registrov (ti so konfigurabilni): v konkretnem primeru so bili uporabljeni: pozitivna delovna energija, negativna delovna energija, navidezna energija po vseh kvadrantih, čas, datum itd.
- števec omogoča zajemanje informacij o implementiranih objektih (registrih): ime objekta, OBIS koda, vsi morebitni pripadajoči elementi (enota, eksponent, decimalno mesto)
- podprto je zajemanje podatkov bremenske krivulje (Load Profile): čas zajetega zapisa, podatki o merjenih veličinah (energija), dogodki (po VDEW standardu)
- podprto je zajemanje podatkov o dogodkih v števcu (Log Book): čas zajetega dogodka, dogodek (po VDEW standardu)

Potek komunikacije (odgovori na zahteve) je razviden iz prilog:

- Vzpostavitev asociacije: Priloga 9.4

8. Sklep

Cilj projekta, ki je bil opisan v magistrski nalogi, je bil razvoj in implementiranje IEC870-5-102 komunikacijskega protokola in DLMS/COSEM specifikacije. Oba protokola je bilo potrebno izvesti na elektronskem industrijskem števcu električne energije.

Programsko kodo protokolov, ki je bila napisana v programskem jeziku C, je bilo potrebno zgraditi modularno in po ISO OSI modelu, saj je na ta način zagotovljena prenosljivost modulov tudi na druge konfiguracije števcev.

Pomemben del razvoja obeh protokolov ni bilo zgolj pisanje kode temveč tudi integracija v elektronski števec. Kodo je bilo potrebno prilagoditi delovanju operacijskega sistema, kjer so se pojavljale številne težave s skladi. Mikrokrmilnik v števcu ima na voljo tudi le 256 kB spomina namenjenega programski kodi. Zaradi tega je bilo potrebno optimirati (krčiti) kodo, še posebno pri DLMS/COSEM specifikaciji. Dosti problemov pri pisanju kode je predstavljala tudi nedosegljivost in kompleksnost specifikacij, ki opisujejo delovanje protokolov.

Sprva so bili moduli testiram na virtualnem elektronskem števcu (gre za simulacijo delovanja realnega števca na osebnem računalniku), sledilo pa je tudi preizkušanje na realnem števcu. Pri IEC870-5-102 protokolu seje pojavila še dodatna težava , ker je bilo testno orodje težko dosegljivo. V ta namen je bila v okviru izgradnje protokola razvita še testna aplikacija PS870. Pri DLMS/COSEM specifikaciji je bila za testiranje uporabljena MeterView 4.0.7 aplikacija. Gre za aplikacijo specifično narejeno za podporo števcev, ki se proizvajajo v Iskraemeco.

Končni rezultat razvoja sta bila dva delujoča komunikacijska protokola namenjena gospodinjskemu in industrijskemu trgu. Oba protokola sta implementirana na elektronskih števcih električne energije (MT860, MT830, MT831) in tudi delujeta. Razvoj na protokolih bo tako potekal še naprej, predvsem zaradi tega, ker se kaže interes trga.

Zagotovo bo naslednji korak v razvoju obeh komunikacijskih protokolov temeljito testiranje. Testiranje na nivoju razvojnega inženirja je bilo že opravljeno, sledi pa še testiranje v testnem laboratoriju, kjer se bo opravila verifikacija, in testiranje števca s protokoli pri sami stranki. V

primeru DLMS/COSEM specifikacije je nujno potrebno izvesti test skladnosti (Conformance Test). Realne možnosti razširjevanja funkcionalnosti protokolov pa se vidi tudi v omogočanju parametriranja števcev (spreminjanje parametrov, ki vplivajo na samo delovanje števca).

Dejstvo je, da ima obstoječi mikrokontroler v elektronskih števcih MT830 in MT831 premajhna oziroma omejena sredstva (predvsem delovnega in programskega pomnilnika za kodo). Ta problem je lahko rešljiv tudi z uvedbo ARM mikrokontrolerjev, ki odpravljajo omenjene pomanjkljivosti. Cilj razvojne skupine industrijskih elektronskih števcev MT830, MT831, MT860 je izdelati števec z vsemi pomembnejšimi komunikacijskimi protokoli (IEC62056-21, DLMS/COSEM, IEC870-5-102 itd.), ki pa bi se vključevali v konfiguracijo števca po potrebi.

9. Priloge

9.1 IEC870-5-102 Inicializacija komunikacije

```
17:07:18:015                                                    REQUEST!
No description
10 49 01 00 4A 16

17:07:18:036                                                    RESPONSE!
No description
10 0B 01 00 0C 16

17:07:19:015                                                    REQUEST!
No description
10 60 01 00 61 16

17:07:19:025                                                    RESPONSE!
No description
10 20 01 00 21 16

17:07:20:015                                                    REQUEST!
Data Class 1
10 4A 01 00 4B 16

17:07:20:022                                                    RESPONSE!
Initialized
ASDU type:      70
ASDU sequence: 1
ASDU cause:     4
ASDU adrs LO:   1
ASDU adrs HI:   0
ASDU rec adrs: 0

End Of Initialization
Cause Of Init: remote reset

68 0B 0B 68 08 01 00 46 01 04 01 00 00 00 02
57 16

17:07:21:015                                                    REQUEST!
Data Class 1
10 6B 01 00 6C 16

17:07:21:026                                                    RESPONSE!
No description
ASDU type:      8
ASDU sequence: 4
ASDU cause:     3
ASDU adrs LO:   1
ASDU adrs HI:   0
ASDU rec adrs: 11

Periodicaly Reset Operational Integrated Totals
Object Number.........1
Object Address:    1
Object Value:      0
Object Number.........2
Object Address:    2
Object Value:      0
Object Number.........3
Object Address:    3
Object Value:      0
Object Number.........4
Object Address:    4
Object Value:      0
Time Of Integrated Total: 09.01.2006  16:07:00

68 26 26 68 28 01 00 08 04 03 01 00 0B 01 00
00 00 00 01 02 00 00 00 00 02 03 00 00 00 00
03 04 00 00 00 00 04 07 10 29 01 06 9F 16
```

```
17:07:22.015                                                    REQUEST!
No description
10 4A 01 00 4B 16

17:07:22.029                                                    RESPONSE!
Data Class 1
ASDU type:      1
ASDU sequence: 1
ASDU cause:     3
ASDU adrs LO:   1
ASDU adrs HI:   0
ASDU rec adrs: 11

Single Point Information
Object Number..........1
Object Address:    238
SPQ: 0
SPI: 1
Time Of SPI: 09.01.2006  16:06:09

68 12 12 68 08 01 00 01 01 03 01 00 0B 00 02
00 24 06 10 29 01 06 86 16
```

9.2 IEC870-5-102 Zajem podatkov razreda 2 (Class 2 data)

```
17:21:01.006                                                    REQUEST!
No description
10 6B 01 00 6C 16

17:21:01.044                                                    RESPONSE!
No description
ASDU type:      8
ASDU sequence: 4
ASDU cause:     3
ASDU adrs LO:   1
ASDU adrs HI:   0
ASDU rec adrs: 11

Periodicaly Reset Operational Integrated Totals
Object Number..........1
Object Address:    1
Object Value:      4068
Object Number..........2
Object Address:    2
Object Value:      2033
Object Number..........3
Object Address:    3
Object Value:      2033
Object Number..........4
Object Address:    4
Object Value:      2032
Time Of Integrated Total: 09.01.2006  16:21:00

68 26 26 68 08 01 00 08 04 03 01 00 0B 01 E4
0F 00 00 01 02 F1 07 00 00 02 03 F1 07 00 00
03 04 F0 07 00 00 04 15 10 29 01 06 67 16
```

9.3 IEC870-5-102 Zajem podatkov razreda 1 (Class 1 data)

```
17:31:07.052                                                    REQUEST!
No description
10 4B 01 00 4C 16

17:31:07.072                                                    RESPONSE!
No description
10 29 01 00 2A 16

17:31:08.052                                                    REQUEST!
No description
10 6A 01 00 6B 16
```

```
17:31:09:029                                    RESPONSE!
Data Class 1
ASDU type:     1
ASDU sequence: 2
ASDU cause:    3
ASDU adrs LO:  1
ASDU adrs HI:  0
ASDU rec adrs: 11

Single Point Information
Object Number.........1
Object Address:   214
SPQ: 0
SPI: 29
Time Of SPI: 09.01.2006  16:31:08
Object Number.........2
Object Address:   0
SPQ: 0
SPI: 27
Time Of SPI: 09.01.2006  16:31:06

68 1B 1B 68 08 01 00 01 02 03 01 00 0B 00 3A
00 20 1F 10 29 01 06 00 36 00 18 1F 10 29 01
06 81 16
```

9.4 DLMS/COSEM Inicializacija komunikacije

```
0 ms elapsed -> HDLC sending (Meter View):
SNRM podatkovni okvir
7e a0 08 20 23 21 93 98 d2 7e

469 ms elapsed -> HDLC received (števec):
UA podatkovni okvir
7e a0 08 21 20 23 73 f9 f5 7e

0 ms elapsed -> HDLC sending (Meter View):
AARQ - prošnja za vzpostavitev asociacije
7e a0 2c 20 23 21 10 8a 29 e6 e6 00 60 1d a1 09
06 07 60 85 74 05 08 01 01 be 10 04 0e 01 00 00
00 06 5f 1f 04 00 00 7e 1f ff ff 99 c8 7e

140 ms elapsed -> HDLC received (števec):
AARE - vzpostavitev asociacije uspela
7e a0 38 21 20 23 30 b7 51 e6 e7 00 61 29 a1 09
06 07 60 85 74 05 08 01 01 a2 03 02 01 00 a3 05
a1 03 02 01 00 be 10 04 0e 08 00 06 5f 1f 04 00
00 7c 1f 00 68 00 07 ec 8e 7e

0 ms elapsed -> HDLC sending (Meter View):
Zajemanje registra o naslovu naprave
7e a0 1a 20 23 21 32 d3 c4 e6 e6 00 c0 01 00 00
01 00 00 2a 00 00 ff 02 00 9d 8a 7e

78 ms elapsed -> HDLC received (števec):
Naslov naprave
7e a0 23 21 20 23 52 0f e2 e6 e7 00 c4 01 00 00
09 10 49 53 4b 2d 4d 54 38 33 30 2f 4c 44 5f 50
55 42 e4 80 7e
```

10. Seznam uporabljene literature

[1] Draft IEC 870-5-102: Telecontrol equipment and systems; Part 5: Transmission protocols, Section 102: Companion standard for the transmission of integrated totals in electric power systems. IEC/TC 57, Nemčija, 26. januar, 2004

[2] IEC 870-5-2: Telecontrol equipment and systems; Part 5: Transmission protocols, Section 2: Link Transmission procedures, CEI/IEC 870-5-2, First edition 1992-2004

[3] Božidar Bratina: OPC kot univerzalni vmesnik v industrijskih aplikacijah, seminarska naloga

[4] R & D: industrijski števci, specifikacije modularnega programja za števce(MODLIB), Jure Artiček, Iskraemeco d.d.

[5] Aleš Podgornik: Razvoj programskih modulov za upravljanje DLMS/COSEM naprav; Magistrsko delo, Univerza v Ljubljani, Fakulteta za elektrotehniko, Ljubljana 2004

[6] DLMS User Association: COSEM Architecture and Protocols (Green Book), DLMS UA 1000-2:2004, Fourth Edition

[7] DLMS User Association, COSEM Identification System and Interface Classes (Blue Book), DLMS UA 1000-1:2002, Fifth Edition;

[8] DLMS User Association: COSEM Identification System (Orange Book), DLMS UA 1000-3:1999, First Edition

[9] G. Kmethy, M. Wisy: Device Language Message Specification, Module: End Users, DLMS/COSEM Training Package, DLMS UA;

[10] G. Kmethy, M. Wisy: Device Language Message Specification, Module: Product Developers, DLMS/COSEM Training Package, DLMS UA;

Spletne strani:

1. Iskraemeco d.d., Merjenje in upravljanje energije, Novice: Podeljena priznanja za "NAJ INOVACIJO Gorenjska 2004", 14.6. 2005,
   http://www.iskraemeco.si

2. DLMS User Association
   http://www.dlms.com

3. UML Rhapsody programsko orodje
   http://www.ilogix.com

Zahvala

Na tem mestu se zahvaljujem vsem, ki ste mi kakorkoli pomagali pri nastajanju tega dela.

Avtor

Izjava

Izjavljam, da sem magistrsko delo izdelal samostojno pod vodstvom mentorja prof. dr. Janka Drnovška.

Ljubljana, 2006 Tomaž Ščuka