Romulus-Mircea TEREBEȘ

COMUNICAȚII MOBILE

Îndrumător de laborator

UTPRESS
Cluj-Napoca, 2019
ISBN 978-606-737-385-1

Romulus-Mircea TEREBEȘ

Comunicații Mobile

Îndrumător de laborator

Editura UTPRESS
Cluj-Napoca, 2019
ISBN 978-606-737-385-1

Editura U.T.PRESS
Str. Observatorului nr. 34
C.P. 42, O.P. 2, 400775 Cluj-Napoca
Tel.:0264-401.999
e-mail: utpress@biblio.utcluj.ro
http://biblioteca.utcluj.ro/editura

Director: Ing. Călin D. Câmpean

Recenzia: Conf.dr.ing. Raul Măluţan
Conf.dr.ing. Nicolae Crişan

ISBN 978-606-737-385-1

Cuprins

Lucrarea 1 - Rețeaua de acces radio GSM. Configurare hardware folosind echipamente Alcatel-Lucent

1.1 Introducere

O rețea de acces este partea unui sistem de comunicații care conectează utilizatorii unui sistem de telecomunicații la furnizorul de servicii, servicii oferite prin intermediul unei rețele nucleu. În terminologie GSM, rețeaua de acces radio este denumită **BSS (Base Station Subsystem)** iar rețeaua nucleu **NSS (Network and Switching Subsystem)**.

Schema bloc simplificată din Figura 1 ilustrează poziția rețelei de acces radio în arhitectura GSM:

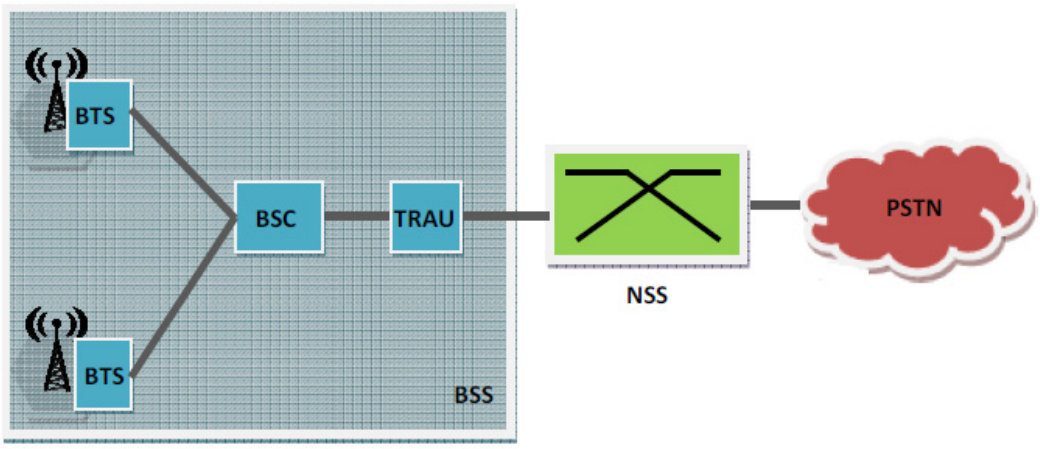


Fig.1 Schema bloc simplificată a unei rețele GSM. BTS- Base Station Transceiver; BSC- Base Station Controller; TRAU- Transcoder and Rate Adaptor Unit

1.2 Funcțiile rețelei de acces radio GSM. Echipamente BSS.

Principala funcție a unui GSM BSS este să asigure canale de semnalizare și canale de trafic bidirecționale pentru transferul vocii/datelor și al semnalizărilor între stațiile mobile și rețeaua nucleu și să asigure suport pentru mobilitatea terminalului prin mecanisme de handover pentru terminale mobile în stare activă. Rețeaua nucleu gestionează stabilirea de apeluri cu alte mobile sau cu rețele externe, realizează tarifarea convorbirilor/sesiunilor de date și implementează mecanisme de suport al mobilității pentru mobile în stare inactivă.

Echipamentele BSS sunt conectate prin interfețe standardizate de specificațiile GSM:

- interfața radio (Um) MS-BTS
- interfața Abis BTS-BSC
- interfața Ater BSC-TRAU
- interața A TRAU-NSS (MSC)

Comunicația între stația mobilă și stația de bază (BTS) are loc prin unde radio, conform unei scheme de acces multiplu de tip FDMA/TDMA în mod FDD (Frequency Division Duplex). FDMA asigură divizarea spectrului de frecvență GSM în 124 de frecvențe purtătoare /sens de comunicație conform principiului ilustrat în fig.2. Operarea în mod FDD presupune utilizarea unei perechi de frecvențe purtătoare

45MHz; o purtătoare pe sens de comunicație.

Echipamentele BTS pot opera de asemenea în banda de 1800 MHz, DCS (Digital Cellular System) – extensia sistemului GSM pentru o capacitate sporită.

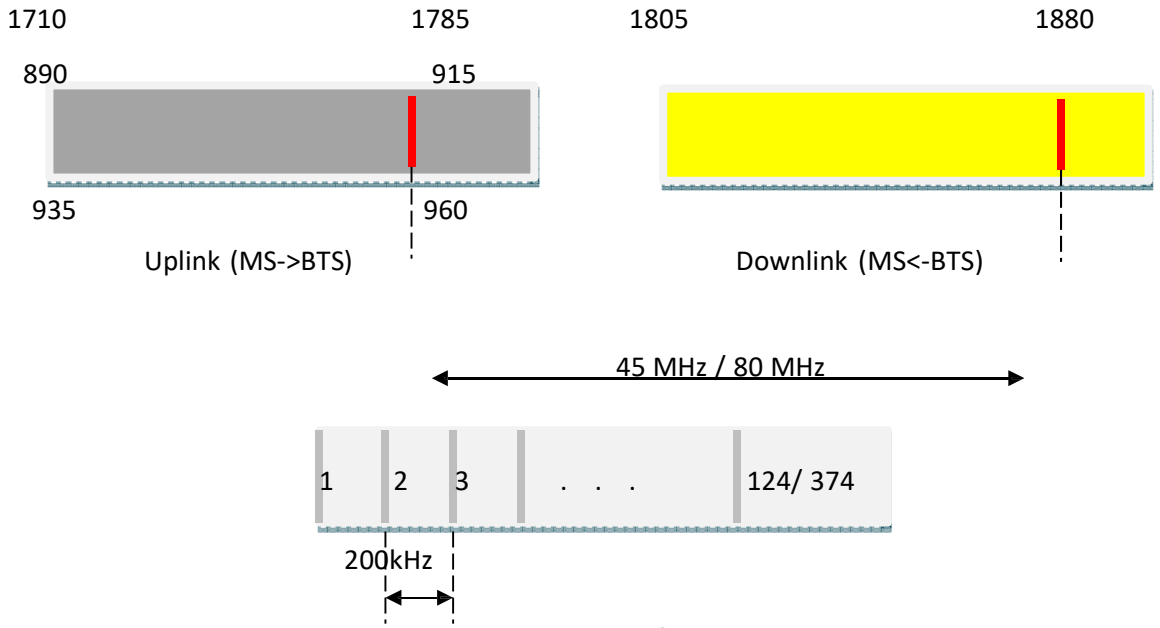


Fig.2 Purtătoare pGSM (primary GSM)/DCS (Digital Cellular System)

Tehnica TDMA presupune organizarea pe fiecare purtătoare de cadre formate din 8 intervale temporale, fiecare interval temporal împreună cu frecvența purtătoare pe care este definit definind un canal fizic GSM.

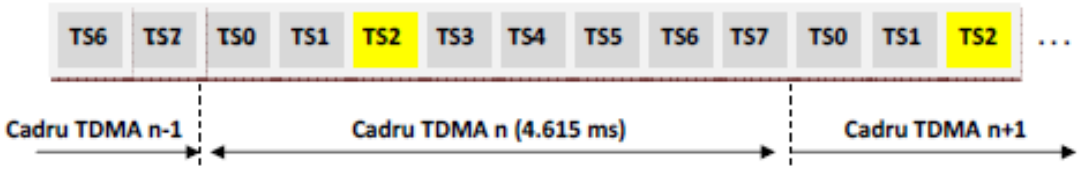


Fig.3 Cadre TDMA GSM

Cadrele TDMA sunt numerotate pentru a permite sincronizarea stațiilor mobile la o structură de multicadre de trafic (26 de cadre TDMA) și de semnalizare (51 de cadre TDMA). Cadrele transmise în direcția uplink sunt decalate cu 3 intervale temporale față de cele din direcția downlink pentru a evita emisia și recepția simultană.

Informațiile transmise pe un canal fizic pot fi voce/date sau semnalizări. GSM utilizează termenul de canal logic pentru a defini tipul de informații transmis pe un canal fizic. Canalele logice folosite de sistemul GSM sunt:

- **canale de trafic (TCH)** – folosite pentru transferul vocii digitizate/datelor
- **canalul de difuzare al celulei (BCCH)** – folosit de către rețea în direcția downlink pentru a

transmite către toate mobilele dintr-o celulă de informații despre aceasta: frecvențe folosite în celulă, frecvențe învecinate, codul ariei de localizare (LAI) etc.

- **canalul de paging (PCH)** – folosit doar în direcție downlink pentru semnalizări în cazul unui apel către stația mobilă
- **canalul de corecție a frecvenței (FCCH)** – folosit doar în direcție downlink pentru a permite stațiilor mobile sincronizarea în frecvență și compensarea deviațiilor de frecvență prin efect Doppler
- **canalul de sincronizare (SCH)** – transportă în direcție downlink informații legate de numărul cadrului TDMA curent și un identificator (BSIC) al frecvenței ce transportă canalele BCCH, FCCH, SCH – frecvență baliză, pe care emisia stației de bază are loc cu putere constantă.
- **canale pentru acces aleator (RACH)** și **canale de confirmare a accesului (AGCH)**- folosite de către stațiile mobile pentru a solicita accesul la rețea (RACH – uplink) și de către rețea pentru a indica alocarea acestora (AGCH- downlink)
- **canale de semnalizări dedicate (SDCCH)** – folosite de către stațiile mobile și de către rețea în ambele direcții pentru autentificare, stabilire parametrii canale de trafic, proceduri de actualizare a localizării, transmisie de mesaje scurte etc.
- **canale de semnalizări lente (SACCH)** – alocate întotdeauna unui TCH sau unui SDCCH. Au un rol extrem de important în GSM și permit: transferul rapoartelor cu măsurători efectuate de către stațiile mobile și transmiterea de SMS-uri pentru mobile angajate într-o convorbire sau sesiune de date (uplink), transferul comenzilor de avansare temporală (TA- timing advance) pentru evitarea interferențelor cauzate de mobilitate utilizatorilor și a informațiilor sistem (lista frecvențelor ce trebuie monitorizate și identificatorii BSIC ai acestora) în direcția downlink.

Operatorul de rețea are posibilitatea configurării modului de mapare a canalelor logice în canale fizice prin multiplexare temporală; uzual maparea canalelor de semnalizare pe frecvența baliză are loc după este ilustrat în figura 4.

Acoperirea radio a unei regiuni geografice se realizează, tipic, în una din configurațiile indicate în Figura 5. Principalul avantaj al utilizării celulelor sectorizate (emisie/recepție din direcții predefinite Fig.5.a) comparativ cu cazul celulelor clasice cu antene omnidirecționale (Fig.5.b), este creșterea capacității fără creșterea semnificativă a interferenței. Fiecare sector se comportă ca o celulă de sine stătătoare cu propriile sale canale iar distanța de reutilizare a frecvențelor poate fi redusă. Efectul final este creșterea capacității prin acoperirea unei arii geografice cu mai multe purtătoare radio față de situația celulelor omnidirecționale. Celulele multibandă (GSM/DCS – Fig. 5.c) oferă avantajul unei capacități sporite prin utilizarea de purtătoare din ambele benzi de frecvență (GSM și DCS) pentru acoperirea aceleaiași arii geografice. Configurația din figura 5.d corespunde acoperirii radio în zone urbane; progapagarea în microcelule are loc în vizibilitate directă (antenele sunt planare și sunt plasate pe fațadele clădirilor) iar BTS-urile asociate sunt configurate să opereze în banda de 1800 MHz (numărul de purtătoare disponibile este mai mare). Acest tip de celule asigură servicii GSM via BSS pentru utilizatori staționari sau care se deplasează cu viteză mică. Celule de tip "umbrelă" acoperă

aceeași zonă geografică, operează uzual în banda pGSM iar rolul lor este minimizarea numărului de handover-uri pentru utilizatori cu o viteză relativ mare de deplasare (la nivel BSC viteza de deplasare a utilizatorilor este cuantificată prin frecvența de apariție a mecanismelor de handover). Antenele corespunzătoare sunt plasate pe acoperișul clădirilor pentru propagare nu neapărat în condiții de vizibilitate directă.

Echipamentul numit BSC (Base Station Controller) este partea ce înglobează "inteligența" rețelei de acces radio GSM. Depinzând ce configurație, un astfel de echipament controlează zeci sau chiar sute de BTS-uri. Principalele funcții ale unui BSC sunt legate de alocarea canalelor radio, recepția și interpretarea rapoartelor cu măsurători construite de stațiile mobile și de BTS-uri, controlul independent sau asistat al mecanismelor de transfer a legăturii (handover intra –BSS și , respectiv, inter-BSC) . Un BSC joacă rolul unui concentrator: comunicațiile recepționate pe un număr relativ mare de interfețe Abis sunt transmise pe un număr sensibil mai mic de legături pe interfețe Ater. Pe lângă funcțiile de control pentru BTS-uri, un BSC implementează funcții de comutare, de semnalizare SS7 cu NSS, de operare și mentenanță pentru configurarea logică a celulelor pe care le controlează și transmiterea alarmelor BSS la nivel OMC (Operation and Maintenance Center). Un BSC memorează local baze de date cu configurația logică a rețelei (liste cu purtătoarele radio din fiecare celulă, parametri pentru saltul de frecvență utilizat pentru contracarea efectelor fading-ului, software-ul instalat în BTS-uri etc). Acești parametri sunt configurați de la nivel OMC în etapa de proiectare a rețelei și pot fi modificați, tot de la acest nivel, funcție de condițiile curente de trafic din celulele supervizate.

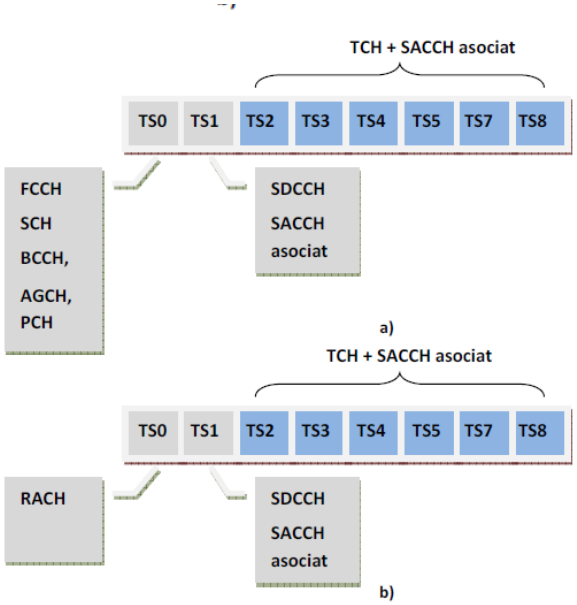


Fig.4 Mapare canale logice în canale fizice. a) Downlink; b) Uplink. Maparea canalelor se face pe o structură de tip multicadru de semnalizări (canalele de control) și de trafic (canalele de trafic și canalele SACCH asociate).

Echipamentul TRAU implementează funcții de transcodare a vocii din format GSM (13kbiți/s) în format PCM

și este adeseori instalat la nivel NSS pentru utilizarea eficientă prin multiplexare a interfețelor Abis și A (4 canale de 13 kbiți/s sunt transmise pe același interval temporal E1 de 64 de kbiți/s). Unii producători de echipamente includ la nivelul TRAU funcții ce asigură evitarea dublei transcodări a vocii 13kbiți/s -> 64kbiți/s->13kbiți/s pentru apelurile mobil-mobil.

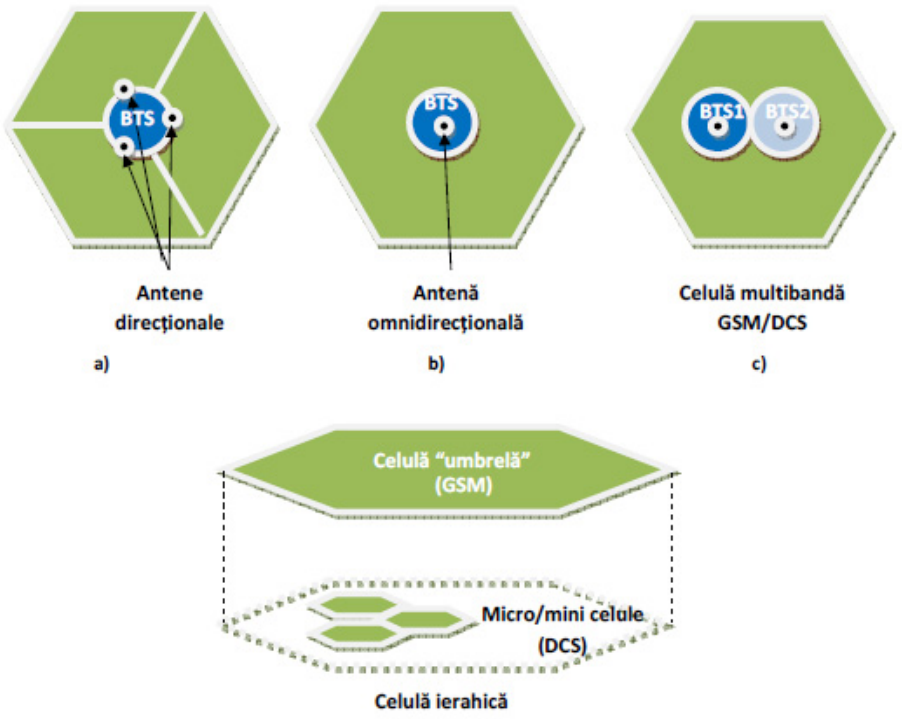


Fig.5 Acoperire radio celulară.
Configurații tipice

Întrebări

1. Indicați pe scurt care sunt principalele funcții îndeplinite de următoarele tipuri de echipamente:

- BTS

- BSC

- TRAU

2. Care este rolul canalului logic BCCH ? Câte canale de acest tip există într-o celulă ?

3. Indicați numărul de legături E1 pentru fiecare dintre configurațiile din figura 5.

4. O stație mobilă trece din stare detașată în stare inactivă. Indicați care sunt canalele logice pe care aceasta le folosește și care sunt informațiile schimbate cu rețeaua pe fiecare canal.

1.3 Schema bloc a platformei de laborator[1]

Platforma de laborator a fost proiectată printr-un proiect comun Alcatel-Lucent/Orange România SA de dotare a principalelor universități din România cu laboratoare moderne de comunicații mobile.

Echipamentele GSM folosite în realizarea configurației hardware din figura 6 sunt, respectiv:

- microBTS-uri de tip Alcatel Evolium 9110 (BTS1, BTS2)
- BSC de tip Alcatel Evolium G2
- surse de alimentare locale de tip Antrice
- reglete de tip Krone cu separare pentru realizarea de bucle locale /separare/ cablare alarme.

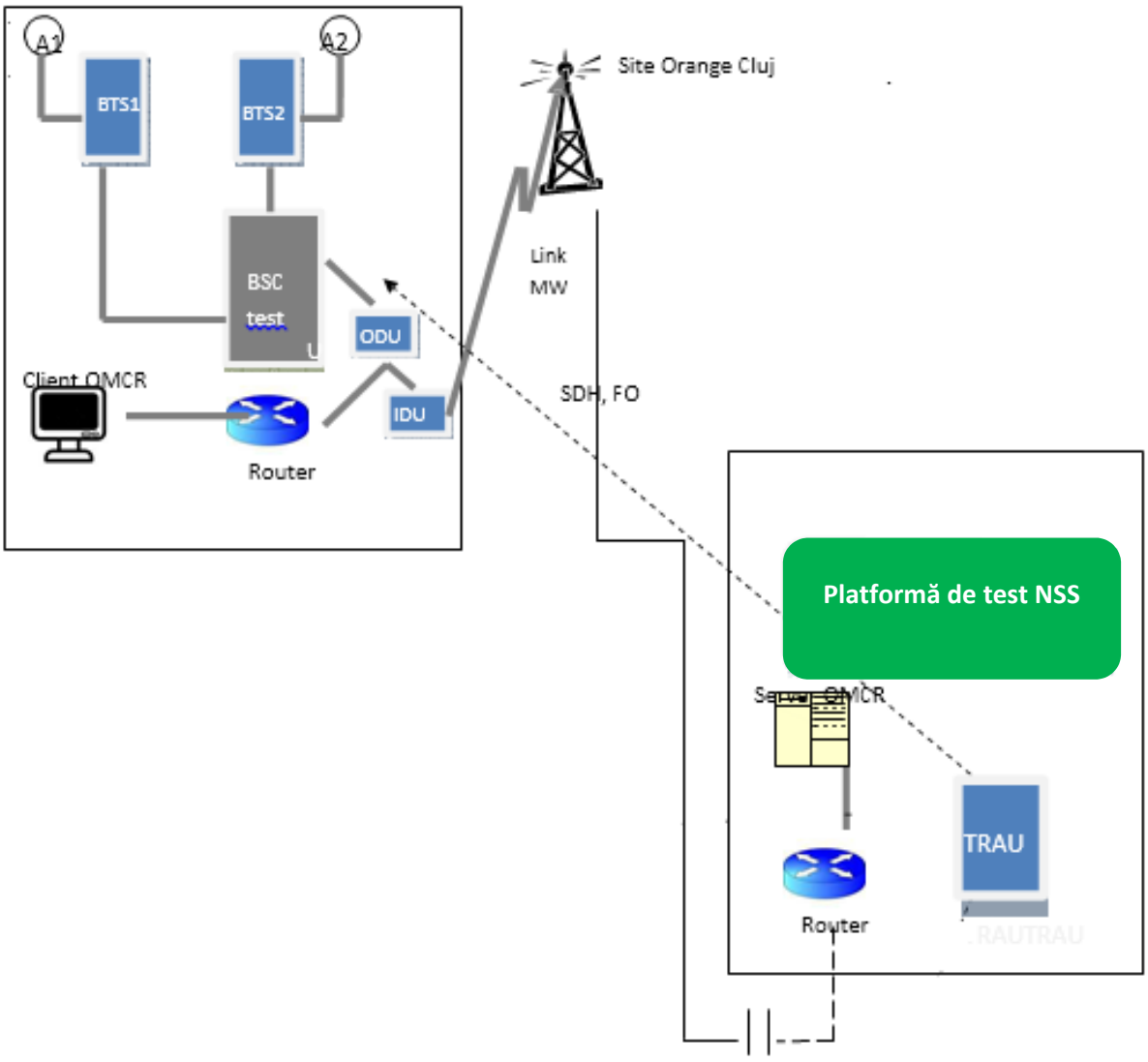


Fig. 6 Schema bloc a platformei de laborator

[1] Dotarea laboratorului de Comunicații mobile din cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a UTCN, cu echipamente 2G a fost realizată cu sprijinul companiilor Alcatel și Orange SA.

1.4 Interfețele BSS

Suportul fizic pentru interfațele Abis și A este multiplexul european primar E1 de 2 Mbiți/s. Canalul temporal TS0 este rezervat pentru sincronizarea de multicadru iar toate celelalte canale pot fi folosite pentru maparea de semnalizări și de date/trafic.

Din punct de vedere logic, echipamentele Alcatel mapează pe Abis următoarele tipuri de informații (fig. 7).

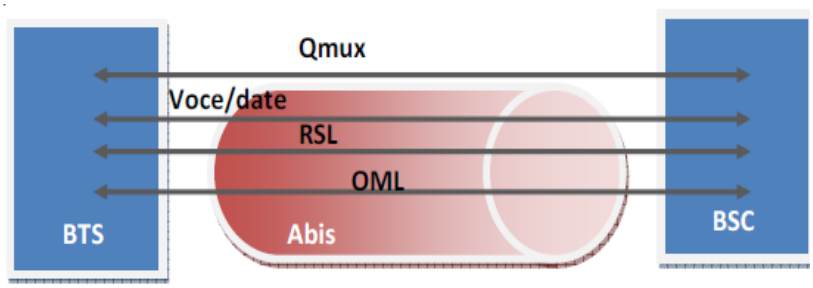


Fig.7 Interfață Abis

RSL- Radio Signaling Link – semnalizări cu un TRE. Numărul de canale RSL este identic cu numărul TRE-urilor cu care este echipat un BTS.

OML – Operation and Maintenance Link – operare și mentenanță

Qmux – configurare și supervizare elemente transmisie (2 biți/TS)

Voce/date – voce 13kbiți/s, date 8/16 kbiți/s mapate. Pe un canal de 64 de kbiți/s sunt mapate 4 canale de voce/date

Din punct de vedere logic, echipamentele Alcatel mapează pe Ater următoarele tipuri de informații:

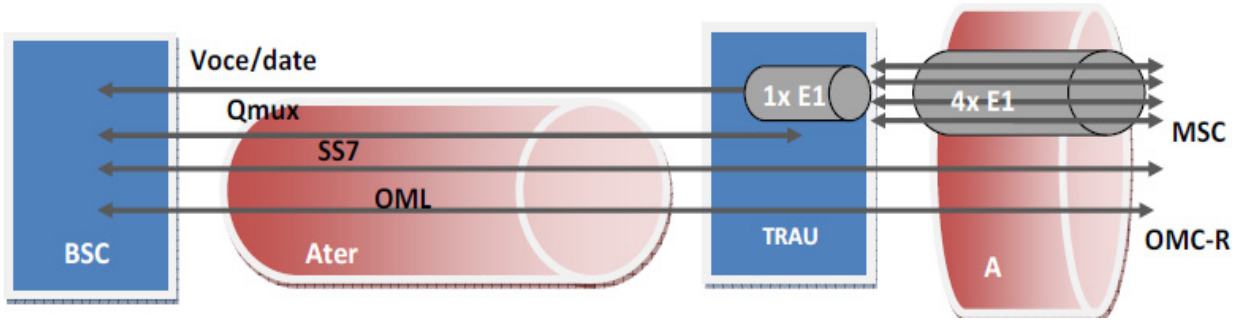


Fig.8 Interfețe Ater și A

1.5 Descrierea funcțională BSC Alcatel Evolium G2 9210

Funcții principale îndeplinite:

- gestiunea resurselor radio (alocare, eliberare canale)
- memorare și întreținere baze de date cu configurația logică a BSS și a software-ului instalat BTS-uri și pe plăcile din structura BSC
- procesare rapoarte cu măsurători de pe interfața radio
- facilități de operare și mentenanță pentru întregul BSS
- comunicare cu OMC-R
- comutație pentru canalele de pe interfața Abis și cele de pe interfața Ater
- controlul semnalizărilor cu MS și BTS via Abis și cu MSC via Ater/A
- măsurători ale performanțelor întregului BSS

Aceste funcții pot fi grupate (conform terminologiei folosite în documentația Alcatel –Lucent) în:

- **funcții de transmisiune** (comutare, multiplexare date/voce, semnalizări, transmisie parametri între elementele BSC, monitorizare elemente transmisie, supervizare)
- **funcții de telecomunicații** (gestiune resurse logice – alocare canale TCH, interpretare semnalizări RACH, parametrii BCCH , intepretare rapoarte cu măsurători, selectare canale pentru handover, semnalizări MSC, paging etc)
- **funcții de operare și mentenanță** (O&M)

Un BSC Evolium este compus din mai multe entități funcționale conectate între ele prin intermediul unei rețele digitale de comutație (**DSN- Digital Switch Network**). Fiecare entitate are o adresă cu ajutorul căreia se stabilesc conexiuni via DSN. Cele mai importante entități sunt:

- **Abis TSU (Abis Terminal Sub Unit)** – asigură conectivitatea cu BTS
- **Ater TSU (Ater Terminal Sub Unit)** – asigură conectivitatea cu TRAU
- **Common TSU** – asigură funcții de tip O&M, gestiune BSC, memorare, încărcare software BSS, configurare elemente transmisie din BSC, stocare bază de date cu configurația BSS, supervizare transmisie.

Orice configurație BSC include cel puțin o unitate Common TSU și una sau mai multe unități Abis TSU și Ater TSU.

Arhitectura simplificată a unui BSC Evolium G2 9120 este reprezentată în figura 9.

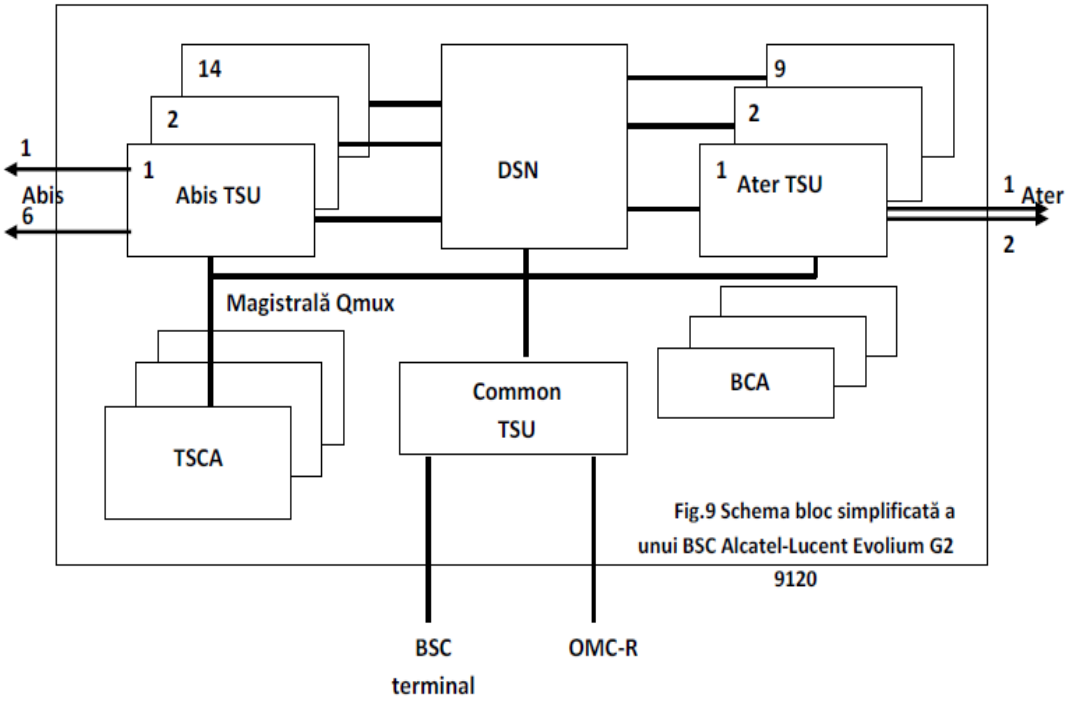


Fig.9 Schema bloc simplificată a unui BSC Alcatel-Lucent Evolium G2 9120

Figura următoare ilustrează configurația folosită în macheta de laborator:

Group switch stage 2	Group switch stage 2	Group switch stage 2
Ater TSU 6	Abis TSU 7	
Abis TSU 4	Abis TSU 5	
Air baffle		
Ater TSU 2	Ater TSU 3	
Group switch stage 1	Abis TSU 1	Clock
TSCA	Common TSU	Group switch stage 2

Fig.10 Cabinet BSC Evolium în configurație 2

Întrebări

1. Folosind documentația tehnică pusă la dispoziție în laborator, indicați pe scurt care sunt principalele caracteristici ale următoarelor module :

- Abis TSU

- Ater TSU

- Common TSU

2. Care sunt echipamentele de transmisiune din BSC?

3. Ce placă BSC gestionează semnalizările SS7?

4. Pe ce placă este stocat software-ul BSS?

5. Fie un scenariu de tip actualizare a localizării. Indicați care sunt echipamentele BSC cu rol activ în această procedură ; pentru fiecare echipament indicat indicați funcțiile pe care acesta le îndeplinește.

6. Fie un scenariu de tip apel MT. Indicați care sunt echipamentele BSC cu rol activ în procedura de paging ; pentru fiecare echipament indicat indicați funcțiile pe care acesta le îndeplinește.

7. Indicați care sunt funcțiile îndeplinite de plăcile cu adresele 117 și respectiv adresa Qmux (QA) 14 .

1.6 Aplicația BSC Terminal. Operații de mentenanță BSC

Icatel-Lucent furnizează pentru echipamentele pe care le produce aplicații ce pot fi folosite pentru configurarea, instalarea, operarea și întreținerea echipamentelor:

- OMC-R – supervizare și control BSS
- BSC terminal – supervizare și control BSC, TC, BTS și echipamente de transmisiune
- BTS terminal – supervizare și control al unui BTS

Aplicația BSC terminal poate fi rulată de pe un terminal conectat la BSC via o interfață serială RS232. Principalele funcționalități oferite de aplicație sunt:

- instalare software BSC – se execută la punerea în funcțiune a unui BSC și presupune copierea tututor fișierelor necesare pe discurile SSD de pe plăcile SYS-CPRC;
- detectare și localizare alarme;
- afișarea configurației hardware;
- modificarea software-ului asociat unei plăci (firmware upgrade);
- modificare parametri logici (definire canale radio, configurare TRE, stabilire parametri pentru saltul de frecvență etc).

Întrebări

1. Folosind regleta Krone identificați unitățile și porturile Abis TSU active. Care este numărul de BTS-uri conectat la BSC?

2. Care este diferența dintre un SBL și un RIT în terminologie Alcatel?

*3. Fisierul **alarme.bdb** listează toate alarmele colectate de către BSS. Analizați alarmele indicate folosind dicționarul de alarme Alcatel*

*3. Fisierul **stari.bdb** listează toate echipamentele BSC care sunt în altă stare decât IT,SOS și NEQ. Explicați pe scurt starea fiecărui echipament și posibilele cauze pentru care se află în această stare. Care este comanda BSC terminal ce permite generarea acestei liste?*

5. Identificați o placă de tip TCU ce gestionează în configurația curentă a BSS legăturile Abis și determinați, folosind aplicația BSC terminal starea în care se află aceasta.
6. Ce funcții îndeplinește placa aflată la poziția 1.3.15 (notație RSS)?

1.7 Descriere funcțională echipamente Alcatel A9110/A9110-E

Echipamentele de tip BTS (Base Transceiver Station) asigură acoperirea radio într-o celulă /sector pe o pereche sau pe perechi multiple de purtătoare în mod FDD. Funcțiile îndeplinite de un BTS Alcatel de tip A9110 sunt ilustrate în Fig. 11.

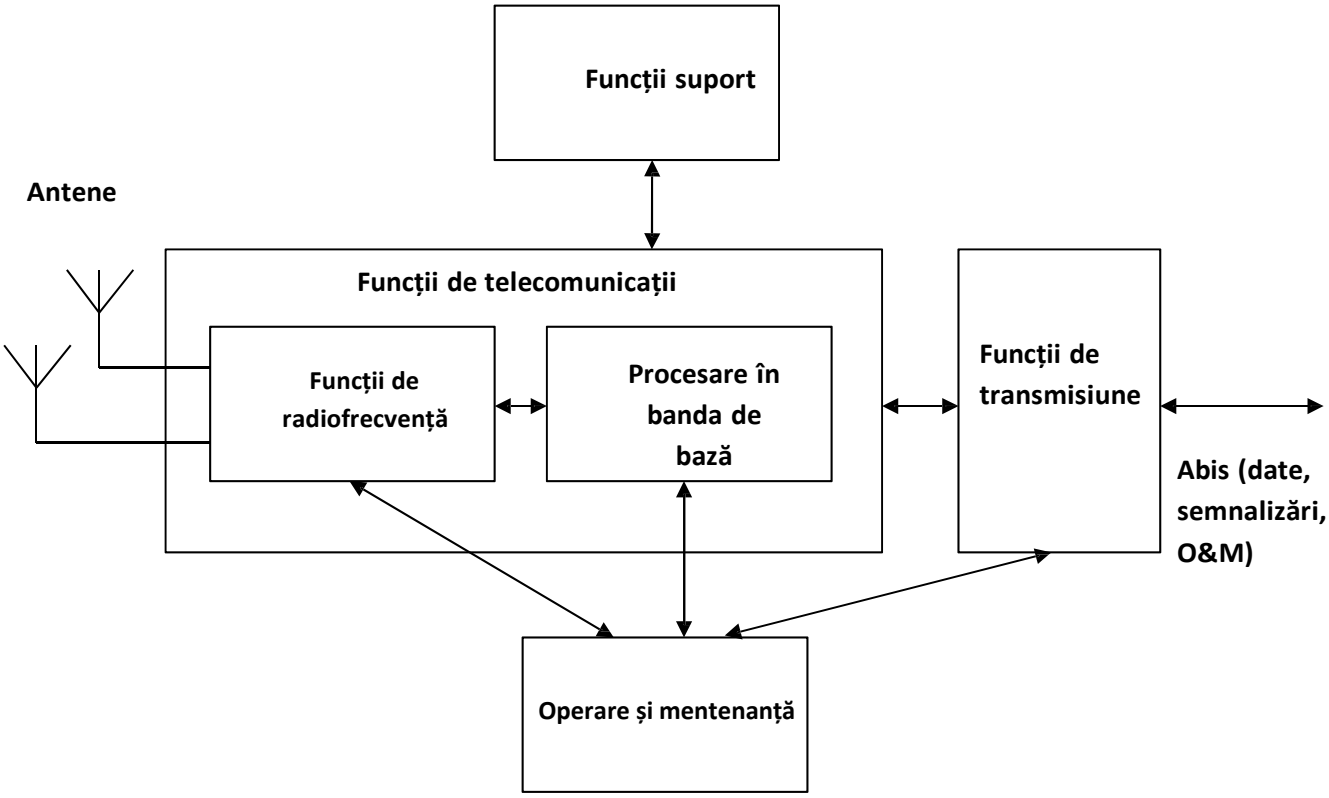


Fig.11 Funcții îndeplinite de un echipament BTS Alcatel A 9110/9120

Conform documentației Alcatel principalele funcții ale unui echipament de tip BTS sunt de următoarele tipuri:

- **funcții de transmisiune** - asigură transferul semnalului vocal, al datelor și al semnalizărilor între un BSC și un BTS, folosind interfața Abis
- **funcții de telecomunicații** –asigură transferul semnalului vocal al datelor și al semnalizărilor între stațiile mobile și BTS pe interfața radio
- **funcții de operare și mentenanță** – configurare, monitorizare BTS
- **funcții suport** – funcții interne ce asigură funcționarea unui BTS (generare semnal de tact, colectare alarme, testare) etc.

a) Funcții de transmisiune

Informațiile ce tranzitează interfața Abis sunt multiplexate într-o structură de tip E1 și, din punct de vedere logic, sunt de următoarele tipuri:

- date, voce
- RSL- Radio Signaling Link
- OML- Operation and Maintenance Link

Funcțiile de transmisiune BTS asigură transferul datelor/semnalului vocal și a semnalizărilor între componentele BTS și echipamentele BSC ce asigură supervizarea acestui transfer (Fig.12).

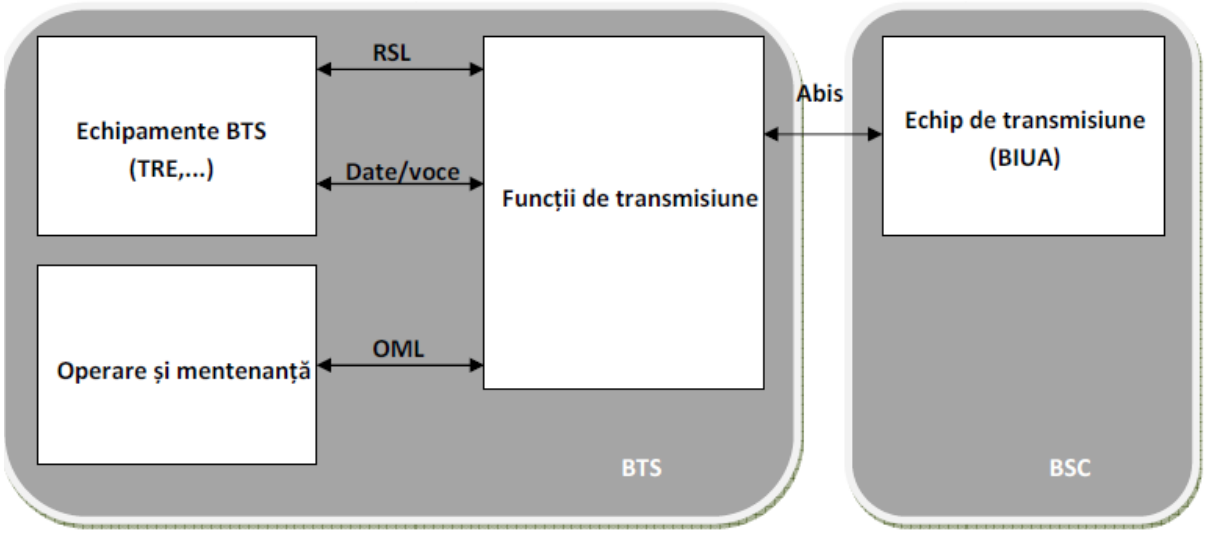


Fig.12 Funcții de transmisiune BTS

Funcțiile de transmisiune Abis permit de asemenea multiplexarea/demultiplexarea datelor pentru interfețe Abis multiple pentru configurații BTS de tip star/chain etc. Figura 13 indică posibile configurații pentru o magistrală Abis ce asigură conectivitatea între un BSC și un BTS cu 4 TRE-uri.

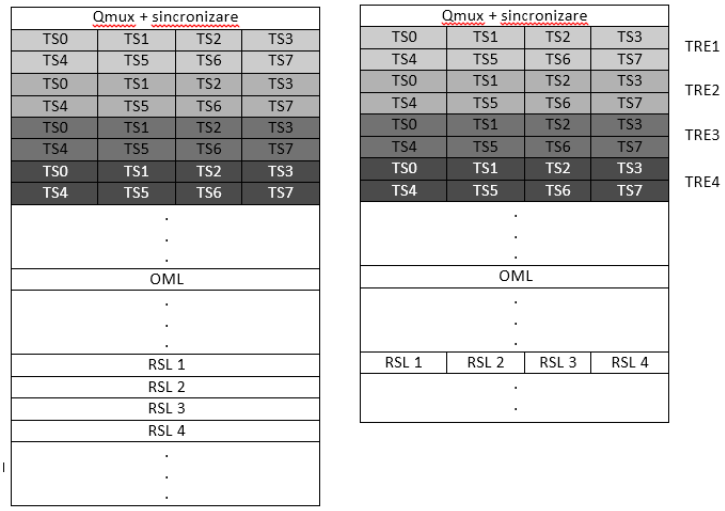


Fig.13 Multiplexare date/voce, RSL, OML pe interfața Abis

b) Funcții de telecomunicații

Sunt implementate pe 2 nivele:

- funcții de procesare în banda de bază - includ toate prelucrările de semnale efectuate de către stațiile mobile exceptând transcodarea vocii
- funcții de radiofrecvență - modulare/demodulare, translatare frecventă, măsurători și control nivel putere etc.

Principalele funcții de telecomunicații îndeplinite de un BTS în banda de bază sunt ilustrate în Fig.14.

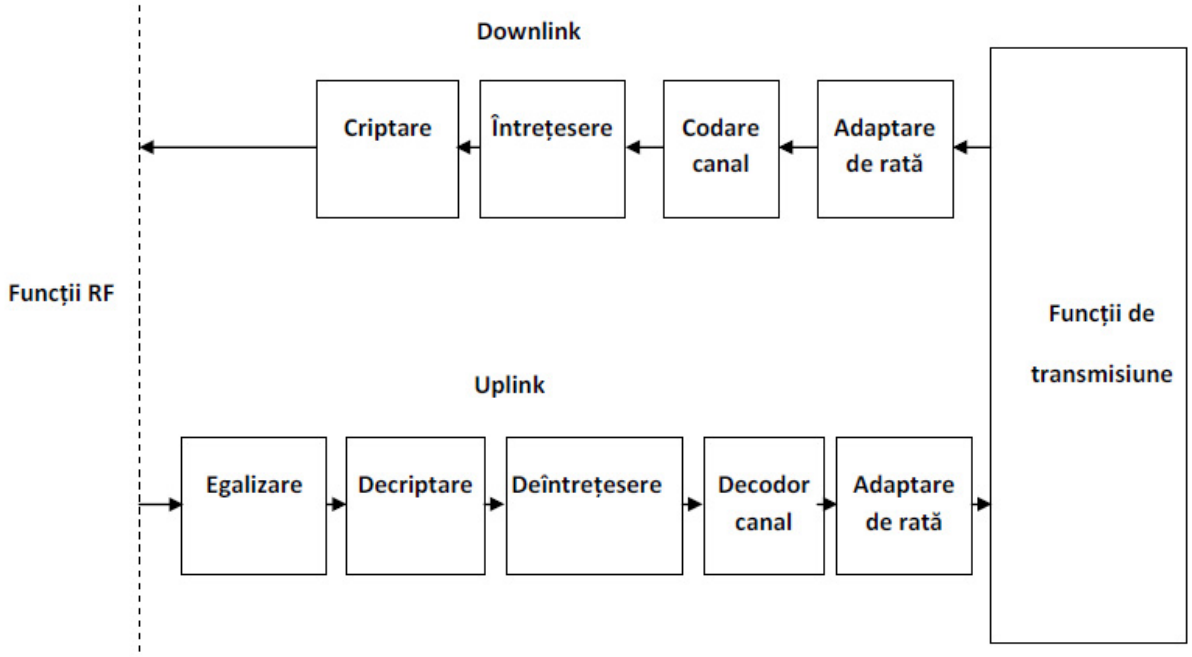


Fig.14 Funcții de telecomunicații BTS – banda de bază

În direcția downlink funcțiile de telecomunicații implementate sunt:

- adaptare de rată – doar pentru transmisii de date; au rolul de a forma cadre V.110 modificate pentru transmisie pe interfața radio GSM (Ex: 16kbps->12kbps pentru serviciul de transport la 9.6kbiți/s)
- codare canal – sunt utilizate coduri corectoare de erori (FEC) pentru servicii de date și tehnici hibride ARQ + FEC pentru servicii de voce
- întrețesere – contracarea efectelor de propagare multicale prin permutarea biților de la ieșirea codorului de canal cu distribuire pe mai multe salva. Adâncimea de întrețesere voce 8 salve normale, date 22 salve, semnalizări 4 salve
- criptare – comunicațiile pe interfața radio sunt securizate în GSM. Tipul de criptare utilizat (cifruri secvențiale *stream cipher*) evită propagarea erorilor de transmisiune de la o salvă la alta și se bazează pe o operație de tip SAU Exclusiv între secvența digitală ce trebuie transmisă și o secvență de criptare binară obținută printr-un algoritm de criptare (A5)

Marea majoritate a funcțiilor de transmisiune în direcția uplink corespund prelucrărilor pereche pentru procesările de semnale efectuate în downlink. Suplimentar un BTS implementează funcții pentru egalizarea canalului pentru contracararea efectelor fadingului selectiv în frecvență.

Funcția de transcodare a vocii este efectuată (distant) de către transcodor (TRAU) pentru a permite utilizarea eficientă a interfețelor Abis/Ater/A cu multiplexarea a 4 canale de voce pe un singur interval temporal (Fig.15)

Funcțiile de radiofrecvență implementate la nivel BTS asigură modularea demodularea semnalelor conform specificațiilo GSM, amplificarea în putere, funcții de tip salt de frecvență și cuplare la sistemul de antene folosind combinere și duplexere.

Amplificatorul de putere este controlabil și permite modicarea nivelului puterii emise funcție de comenzi primite de la BSC (pentru canale de trafic TCH). Canalul de difuzare al celulei (BCCH) este emis cu o putere constantă pentru a permite efectuarea măsurătorilor de putere la nivelul stațiilor mobile pornind de la aceeași referință.

Un TRE poate opera la o frecvență fixă sau pe frecvențe multiple în cazul în care se implementează salturi de frecvență (frequency hopping). GSM folosește salturi de frecvență lente (slow frequency hopping) ; rezoluția pentru saltul de frecvență pentru un TCH fiind dată de un interval temporal. Lista frecvențelor pe care se efectuează saltul este inclus într-o listă (Mobile Allocation MA – în specificațiile GSM). Singurele canale logice ce pot fi supuse opțional saltului de frecvență sunt TCH și SDCHH; pentru toate celelalte canale frecvența este fixă (frecvența baliză). Pentru direcția downlink saltul de frecvență are loc fie prin comutarea unui TCH pe diferite frecvențe purtătoare (diferite TRE-uri cu frecvențe de emisie constante) fie prin modificarea frecvenței purtătoare pe care operează un TRE cu ajutorul unor oscilatoare programabile. Diagrama funcțională din figura 6 corespunde ultimului caz descris mai sus; o astfel de configurație permite ca numărul de frecvențe pe care se efectuează saltul să fie mai mare decât numărul de TRE-uri . Frecvența/ele pe care operează oscilatoarele programabile sunt setate prin comenzi O&M primite de la nivel BSC sau

c) Funcții de operare și mentenanță. Funcții support

Toate entitățile BTS sunt controlate la unitatea de operare și mentenanță iar funcțiile O&M sunt divizate între BSC și BTS. Toate modulele BTS sunt adresabile intern via unitatea de operare și mentenanță (OMU) care poate fi interfațată cu o aplicație dedicată BTS terminal via RS232 prin care se pot iniția comenzi asupra echipamentelor din BTS. Funcțiile de operare și mentenanță principale sunt:

- descărcare software de aplicație și fișiere de date de pe BSC
- configurare și reinițializare module BTS conform fișierelor descărcate
- funcții de tip Remote Inventory ce permit detectarea și raportarea modulelor BTS instalate precum și a interconexiunilor acestora la nivel OMC
- extensii hardware – adăugarea de module și configurarea ulterioară a acestora via BSC
- reinstalare software fără întreruperea serviciilor de telecomunicații

- detectare alarme și reinițializare software/hardware prin funcții de tip reset
- generare/sincronizare și distribuire semnal de tact către toate modulele BTS

Întrebări

1. Care este diferența dintre funcțiile de telecomunicații și funcțiile de transmisiune ?

2. Reprezentați grafic o structură posibilă a unei interfețe Abis ce interfațează un BTS cu 4 TRE-uri cu un BSC. Legăturile RSL sunt de 16 kbiți/s și sunt multiplexate static. Care este numărul maxim de TRE-uri ce ar putea fi conectate în acest caz? Care ar fi numărul maxim de convorbiri dacă acoperirea radio este omnidirecțională? Dar dacă acoperirea radio este realizată cu 3 sectoare?

3. Care sunt principalele avantaje ale utilizării unor combinere și duplexere?

4. Reprezentați o diagramă simplificată care să ilustreze un salt de frecvență ciclic pentru 1 canal TCH într-un BTS echipat cu 4 TRE-uri. Numărul de frecvențe din lista MA este presupus ca fiind egal cu 9.

5. Fie un apel de tip M . Care sunt resursele logice și fizice folosite în rețeaua de acces (BSC, BTS și interfețele terestre/radio) folosite pentru a semnala un apel către stația mobilă apelată?
6. Care sunt principalele module ale unui Micro BTS Alcatel?

6. Indicați pe scurt cu referire la paragraful cum sunt distribuite funcțiile unui BTS pe modulele unui Micro BTS Alcatel 9110

8. Folosind regleta Krone determinați care este topologia de interconectare a celor 2 BTS-uri la BSC

1.8 Aplicația BTS terminal

Aplicația BTS terminal rulează pe un PC/laptop și permite interfațarea acestuia cu un BTS pentru efectuarea de comenzi de tip O&M. Aplicația permite de asemenea inițierea de comenzi pentru BTS-ul la care se conectează, vizualizarea alarmelor, initiațializarea modulelor hardware și a configurației logice.

Întrebări

1. Utilizați meniul Help al aplicației BTS terminal pentru identificarea comenzilor ce pot fi inițiate pentru a determina:
- frecvențele utilizate de către BTS
-modul de alocare a intervalelor temporale pe interfața Abis
- modul de mapare și intervalele temporale folosite pentru RSL-uri
- frecvența ce transportă BCCH
- modul de criptare implementat curent pe BTS
- dacă cele 2 antene din laborator corespund unei configurații cu diversitate în spațiu
- parametrii de indentificare a BTS la nivel MS (NCC, BCC).
- modul de alocare a canalelor pentru fiecare MTRE

Bibliografie

[1] Alcatel BSS. Evolium BSC/TSC overall description, Alcatel University.
[2]. Evolium Base Station Subsytem description, Alcatel University.
[3]. ALCATEL 9120 Base Station Controller. Maintenance**,** Alcatel University.
[4]. Alcatel 9120 Base Station Controller. Description, Alcatel University
[5]. Alcatel EVOLIUM BTS A9100/A9110/A9110–E Functional Description
[6]**.** Alcatel University - Evolium™ Mobile Radio Solutions, Alcatel 9110 Micro-BTS Description
[7]. Alcatel University - Evolium™ Mobile Radio Solutions, Alcatel 9110 Maintenance
[8] Comunicații mobile, notițe de curs, http://ares.utcluj.ro

Lucrarea 2 Setul de comenzi AT GSM. Serviciul SMS

2.1 Introducere

Suplimentar serviciului de telefonie GSM permite utilizatorilor folosirea de servicii de transmisii de date. Acestea pot fi accesate prin conectarea la o staţie de telefonie mobilă GSM a unui computer extern, a unui laptop, sau a altor echipamente (**TE- Teminal Equipment**). Folosind aceste servicii un utilizator poate face schimb de date cu reţele dedicate: PSPDN (rețele de date cu comutație de pachete– ex. X. 25, Internet), CSPDNs (rețele de date cu comutație de circuite), ISDN, poate trimite/primi faxuri către/de la utilizatori conectaţi la PSTN (reţeaua de telefonie publică comutată). Standardul GSM permite [2]:

- controlul terminalelor mobile pentru configurarea parametrilor asociați transferului de date.
- utilizarea terminalelor externe pentru configurarea, trimiterea și citirea de mesaje scurte prin intermediul terminalelor mobile.

Controlul terminalelor mobile se realizează printr-un set dedicat de comenzi în format alphanumeric, numit în specificații ***comenzi AT*** (**ATtention**), care se transmit printr-o interfaţă serială fizică (RS232) sau emulată prin USB, Bluetooth , IrDA etc.

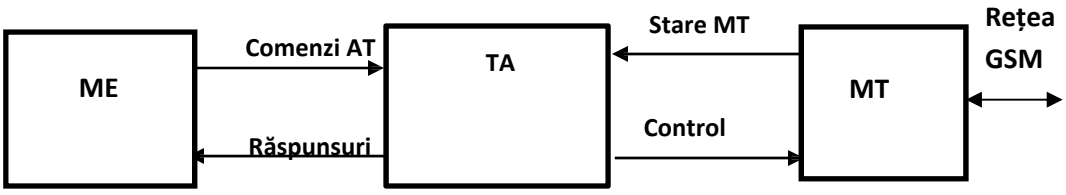


Fig.1 Configurație tipică pentru controlul extern al unui modem GSM

Conform specificaţiilor GSM echipamentele mobile pot include un dispozitiv numit generic **TA erminal Adaptor**). Acesta are rolul de a asigura conversia din setul de comenzi AT în comenzi directe upra părții radio a stațiilor mobile **MT** (**Mobile Termination**). Implementarea funcțiilor de tip TA a fost itial opțională pentru stații mobile GSM, devenind ulterior obligatorie pentru stații mobile de generație ai nouă.

GSM implementează setul de comenzi AT definit de recomandarea ITU-T V. 25ter [1] şi comenzi AT suplimentare.

2.2 Tipuri de comenzi AT

Principalele tipuri de categorii de comenzi AT sunt listate în cele ce urmează.

A. Comenzi AT generale

Comenzile de acest tip permit identificarea MT: producător, versiune software, revizuire, IMEI. IMSI. Acestea pot fi de asemenea utilizate pentru a selecta un set de caractere pentru echipamentul de date (TE).

B. Comenzi pentru controlul apelurilor

Acestea permit selectarea tipului de număr utilizat pentru apelare (national/international), apelarea dintr-o agendă telefonică, selectarea tipului de apel (voce/date/fax), selectarea modului de transfer al datelor în interiorul rețelei PLMN (sincron/asincron, debit, transparent sau netransparent), configurarea unor parametri de control pentru apeluri (ex: configurarea mesageriei vocale), generarea de rapoarte, configurarea/verificarea parametrilor HSCSD pentru terminale compatibile, configurarea parametrilor pentru compresia de date conform standarului V. 42bis etc.

C. Comenzi pentru controlul serviciilor oferite de rețea

Acest tip de comenzi permit afișarea MSISDN, configurarea și verificarea serviciilor suplimentare, înregistrarea în retea și selectarea operatorului.

D. Comenzi pentru controlul MT și verificarea stării acestuia

Comenzile AT care intră în această categorie permit în principal inițierea de acțiuni la nivelul MT pentru citirea/scrierea informații din locații diferite, configurarea ceasului în timp real, accesarea informațiilor de pe cartela SIM, verificarea calităţii semnalului recepţionat.

E. Comenzi pentru gestionarea erorilor MT

Comenzile specifice din această categorie permit activarea/dezactivarea de funcții de raportare a erorilor de pe MT.

G. Comenzi GPRS/comenzi pentru accesarea serviciilor cu comutație de pachete [6]

Aceste tip de comenzi permit setarea parametrilor pentru sesiuni de date cu comutație de pachete.

GPRS utililizează termenul de context **PDP (Packet Data Protocol)** pentru caracterizarea parametrilor unei sesiuni de date. Un astfel de context include o adresă IP și numele unui punct de acces care rețeaua de date externe la care utilizatorul dorește să se conecteze.

F. Comenzi AT pentru serviciul de mesaje scurte

Această categorie regruprează comenzile specifice pentru trimiterea și recepționarea de SMS-uri, setarea parametrilor pentru acest serviciu (format utilizat, număr centru SC etc).

2.3 Sintaxă

Toate comenzile utilizează subșirul AT ca prefix, eventualele spații sunt ignorate iar comenzile trebuie să se finalizeze prin caracterul <CR> (carriage return).

Formatul utilizat pentru **comenzile simple** este:

AT<comandă>[=][<parametru1>, <parametru2>]<CR>

<comandă **>** - șir care identifică tipul de comandă
[ ] – secvența între aceste caractere delimitatoare este opțională
- parametrii sunt opționali, dacă comanda acceptă mai mulți parametri aceștia sunt separați prin caracterul “,”.

Exemple: ***ATP<CR>*** *- selectarea modului de apelare pulse,* ***ATE0<CR>*** *– suprimare ecou (caracterele introduse de la tastatură și trimise către MT nu sunt afișate în consolă).*

Sintaxa folosită pentru **comenzi AT extinse** este următoarea:

AT+<comandă>[=][<parametru1>; <parametru2>]<CR>

Caracterul **"+"** este utilizat pentru toate comenzile din această categorie.

Exemplu: AT+CSCS<CR> = "GSM" *(transmiterea către MT a informațiilor privind setul de caractere folosind pe echipamentul DTE, GSM în acest caz)*

Mai multe comenzi extinse pot fi transmise simultan. Sintaxa folosită în acest caz este:

AT+<comandă1>[=][<parametru1>];+<comandă2>[=] [<parametru2>]<CR>

2.4 Comenzi AT de citire și test

Toate comenzile care acceptă parametri au asociate **comenzi de citire** (*eng. read*) ce se folosesc pentru a verifica valorile curente ale parametrilor. Sintaxa unei comenzi de citire este:

AT+<comandă>?<CR>

Exemplu: **AT+CSCS?<CR>** (interogare pentru citirea setului de caractere folosit curent)

Comenzile AT extinse au asociate și **comenzi de test** (*eng. test*) care sunt folosite pentru a verifica dacă comenzile sunt implementate și de a returna valorile posibile ale parametrilor. Sintaxa folosită pentru astfel de comenzi este:

AT+<comandă>=?<CR>

Răspunsurile la comenzi AT sunt retunate în mod text și pot fi de mai multe tipuri:

-"OK" – indică faptul că o comandă a fost executată cu succes

- răspunsuri extinse în sintaxa:

<CR><LF>+<comandă>:<valoare><CR><LF>OK<CR><LF>

<valoare> - rezultat returnat; dacă există mai multe valori returnat acestea sunt separate prin caracterul ","

<LF> caracter line feed

<comandă>- șir care indică care a fost comanda AT care a generat rezultatul.

Dacă o comandă nu implementată sau dacă sintaxa utilizată pentru trimitere e incorectă răspunsul MT via TA include caracterele “ERROR” .

Exemple

Un răspuns posibil la o comandă de tipul ***AT+CSCS=?<CR>*** *poate fi :*

+CSCS: ("GSM","UCS2")
<CR>
OK
<CR>

Răspunsul la aceeași comandă în mod citire mode ***AT+CSCS?<CR>*** *poate fi:*

<CR>
+CSCS:”GSM”
<CR>

Atunci când comanda este folosită pentru setarea setului de caractere ***AT+CSCS=”GSM”<CR>*** *, în ipoteza în care acesta este implementat, răspunsul este format astfel::*

<CR>
OK
<CR>

2.5 Principalele tipuri de comenzi AT generale

Comandă	Descriere	Răspuns	Observații
AT	verifică dacă MT este funcțional	OK +CME:error	
AT + CGMI	afișează numele producătorului echip. MT	<producător MT> OK	
AT + CGMM	afișează numele generic/tipul MT	<model> OK	
AT + CGMR	afișează versiunea software de pe MT	<versiune software> OK	
AT + CGSN	afișare număr serial echipament MT (IMEI)	<IMEI> OK	
AT + CSCS=<ch. set> AT + CSCS=?	setare set caractere care va fi folosit de MT afișează valorile posibile	OK	
AT + CIMI	vizualizare identficator GSM (IMSI – International Mobile Subscriber Identity)	<IMSI> OK	IMSI este un identificator asociat cartelei SIM și abonamentului asociat
AT+WS46=<n> AT + WS46=?	Permite selectarea tipului de rețea mobile	OK	Comanda de test afișează valorile posibile (ex: 12- GSM, 22 UTMS, 29 GSM și UMTS etc)

Comandă	Descriere	Răspuns	Observații
AT+GCAP	Permite vizualizarea funcțiilor implementate la nivel de TA	<listă funcții>	+FCLASS- fax + CGSM + DS – servicii de date

2.6 Principalele tipuri de comenzi AT pentru controlul apelurilor

Comandă	Descrierer	Răspuns	Observații
ATD<dialstring>;	Inițiează un apel de voce	OK NO DIAL TONE ERROR BUSY NO CARRIER	
ATD<dialstring>	Inițiează un apel de date	Unul dintre răspunsurile de mai sus+ CONNECT	Un serviciu suport trebuie selectat în prealabil
ATD><Name>;	Apel din memorie	Similar cu ATD<dialstring>;	Utilizatorul <Name> memorat în prealabil într-o locație de memorie este apelat
ATD>n;	Apel indexat din memorie	Similar cu ATD<dialstring>;	Numărul plasat la indexul n este apelat
ATA	Răspuns apel	OK CONNECT CONNECT <parameters> NO CARRIER ERROR	
ATH	Terminarea unui apel	OK ERROR	
AT+CMOD=n	Setare mod apel	OK	Doar voce/date, voce și date alternat etc
AT+CMOD=?			Valori implementate Un răspuns tipic poate avea forma: +CMOD: (0-2) OK 0- single mode 1- alternate voice/fax 2- alternate voice/data
AT + CRC = n	Control al modului de afișare pentru un apel primit	OK	AT+CRC=0 RING AT+CRC=1 +CRING:<tip de apel> *Exemplu:* +CRING: VOICE

Comandă	Descrierer	Răspuns	Observații
AT+CBST= [<speed>[,<name>[,<ce>]]] AT+CBST=?	Selectarea unui serviciu suport	OK	Doar apeluri de date <speed>0-selectare automată, …4 -4800bps, …7-9600 bps <name> 0- asincron 1- sincron <ce> 0- transparent 1- netransparent *Exemplu:* AT+CBST=0,0,0 Indică valorile implementate [1]
AT+CHUP	Terminare apel	OK	Pentru servicii alternate voce/date
AT+CSTA=<type> AT+CSTA=?	Selectare format numere de telefon	OK	<type> 145, 129 pentru numere în format international și, respectiv, național *Exemplu*: +CSTA (129,145)
ATT/ATP	Formare ton de apel de tip TONE/PULSE	OK	
AT+CRLP	Setări mod netransparent pentru servicii de date	OK	
AT+DS= [<direction>[,<compression_negotiation>[,<max_dict>[,<max-string>]]]] AT+DS=?	Setare parametri compresie de date (LZW)	OK	Compresia de date este posibilă atât în uplink cât și în downlink *Exemplu:* +DS:(0-3),(0,1),(512-2048),(6-32)
AT+CHSD	Afișare parametrii HSCSD	+CHSD: <mclass>,<maxRx>, <maxTx>,<sum>,<codings> OK	mclass- clasă multislot maxRx-număr maxim de intervale temporale pe direcție DL maxTx-număr maxim de intervale temporale pe direcție UL sum – număr total de interval temporale
AT+CHST =[<wAiur>[,<wRx>[,<topRx>[,<codings>]]]]	Setare parametrii pentru transmisii HSCSD	OK	

2.7 Principalele tipuri de comenzi pentru controlul serviciilor oferite de rețea

Comandă	Descriere	Răspuns	Observații
AT +CNUM	Număr telefon	MSISDN's	Afișare MSISDN

Comandă	Descriere	Răspuns	Observații
AT+CREG=<n> AT+CREG ?	Înregistrare în rețea	OK	 Comanda de citire poate fi folosită pentru localizare Dacă n=2 răspunsul AT+CREG? este: +CREG: <n>,<stat>,<lac>,<ci> OK stat – stare înregistrare lac- Location Area Code ci- Cell Identity
AT+COPS=<mode>, [<format> [, <oper>], [, <AcT>]]] AT+COPS?	Selectare operator		mod – automat (0), manual (1) format – lung(0)/scurt(1)/numeric(2) pentru nume operatori oper – code of the operator (MCC/MNC) AcT – tip rețea (GSM/GPRS, UMTS et) citire operator curent și afișare pe TE
AT+CLIP/CLIR=<n>	Calling line identification presentation/restriction		<n>=0/1 permite dezactivarea/activarea serviciului suplimentar

2.8 Principalele tipuri de comenzi pentru controlul MT și verificarea stării acestuia

Comandă	Descriere	Răspunsuri	Observații
AT +CSQ	Afișare nivel semnal recepționat și probabilitate de eroare	+CSQ: <rssi>,<ber>	<rssi>- received signal strength indicator **0**: -113 dBm sau mai puțin; 1: -111 dBm **2 - 30**: -109 –53 dBm **31**: mai mare de-51dBm **99**:valoareanu poate fi determinată **<ber>**: valoare BER **99**: valoarea nu poate fi determinată
AT + CFUN=<n> AT+CFUN?	Setarea funcțiilor la nivel de MT	OK	<n> valori indexate

Comandă	Descriere	Răspunsuri	Observații
AT + CKPD=<numeric string>	Emularea unei tastaturi	OK	*Example:* AT + CKPD="*#06#" – displays IMEI on the MS screen
AT +CBC	Afișare nivel baterie	+CBC: <bsc>,<bcl>	<bsc>- prezență baterie, încărcător <bcl>- nivel încărcare[%]
AT+CPBS=<string>	Selectarea unei locații de memorie pentru stocare	OK	Exemple de valori pentru <string> SM- card SIM ME – terminal
AT+CPBS=?	Afișarea locațiilor de memorie posibile	+CBPS ("ME","SM","DC","LD","FD", "MC","RC")	
AT + CPBR= <index1>[,<index2] AT+CPBR=?	Aăfișare numere de la o locație selectată Afișează dacă comanda este implementată	AT+CPBR= <index1>,<number>, <type>,<text> +CPBR: <index2>,<number>,<type>, <text>] OK AT+CPBR= <list of indexes>),[<nlength>],[<tlength>]	<type> 129 145 <name> nume <nlength> - lungime maximă pentru stocarea numărului length for <number> <nlength> - lungime maximă pentru stocarea câmpului <text" <text>
AT+CPBW=<index> [,<number>[,<type> [,<text>]]]	Scriere la o locație în memorie	OK	Parametrii sunt identici cu cei descriși la AT+CPBR

2.9 Principalele tipuri de comenzi GPRS și pentru accesarea serviciilor cu comutație de pachete

Comandă	Descriere	Răspuns	Observații
AT++CGDCONT= [<cid> [,<PDP_type> [,<APN> [,<PDP_addr> [,<d_comp> [,<h_comp> [,<pd1> [,…[,pdN]]]]]]]]	Definește un context PDP	OK ERROR	<cid>- identificator context <PDP_type> - tipic IP pentru conexiuni internent <APN> - Access Point Name – nume punct de acces <PDP_addr> - IPv4 IPv6;

Comandă	Descriere	Răspuns	Observații
AT+CGATT= [<state>]	Atașare și detașare explicită de la rețeaua GPRS	OK ERROR	<state> 1 pentru atașare, 0 pentru detașarea de la rețea.
AT+CGACT=[<state> [,<cid>[,<cid>[, …]]]]	Activarea unui context definit anterior	OK ERROR	*Exemplu:* AT +CGACT=1,1
AT+CGDATA=[<L2P> ,[<cid> [,<cid> [,…]]]]	Stabilirea unei sesiuni de date	CONNECT ERROR	<L2P> - protocol de strat 2 ce va fi folosit, echivalentă cu ATDT*99#
AT+CGPADDR= [<cid> [,<cid> [,…]]] +CGPAD DR: <cid>,<PDP_addr >	Afișare adresă IP		Adresa IP este alocată de rețea

Întrebări

1. *Ce sunt comenzile AT?*
2. *Care este scopul utilizării comenzilor AT de citire? Care este sintaxa asociată?*
3. *Care este scopul utilizării comenzilor AT de test? Care este sintaxa asociată?*
4. *Care sunt la comenzile care pot fi utilizate pentru a afla și afișa pe un dispozitiv distant numărul serial al echipamentului/versiunea de software instalată/modelul/identitatea GSM?*
5. *Care este comanda care poate folosită pentru apelarea unui număr? Cum se face distincţia între un apel de voce şi unul de date?*
6. *Care sunt posibile locatii de stocare pe un telefon mobil?*

7. Care sunt parametrii unui serviciu suport (BS)? Indicați cum poate fi un astfel de serviciu selectat.

8. Care este comanda care poate oferi informații de localizare în interiorul unei rețele GSM PLMN? Faceți aprecieri cu privire la precizia informațiilor de localizare obținute.

9. Descrieți modul în care lista operatorilor dintr-o regiune oarecare poate fi afișată pe un echipament de tip TE.

10. Care este secvenţa de la comenzi care pot fi utilizate pentru a afişa numerele stocate pe o cartelă SIM?

11. Care este secvenţa necesară de comenzi AT pentru activarea unei sesiuni de date într-o reţea GPRS?

12. Utilizând informațiile din referința [1] indicați dacă este posibilă transmiterea unui fișier de date în timpul unui apel vocal. Care este secvența de comenzi AT care ar trebui să fie utilizată?

2.10 Exerciții

Terminalele de tip MT utilizate în lucrarea de laborator pentru testarea comenzilor AT sunt de tip MT2, prezentând o interfață serială ce permite conectarea unui terminal extern. Trimiterea unor comenzi AT se face folosind o aplicație de tip hyperterminal (*Tera Term*).

1. Configurați o legătură serială pe portul serial listat în "Device Manager-> Modems".
2. Folosind aplicația Tera Term, inițiați comenzile AT necesare pentru afișarea versiunii de software, a tipului echipamentului mobil, a identificatorului IMSI precum și a tipului de rețea mobile în care poate acesta opera. Notați rezultatele obținute.
3. Care sunt serviciile suport implementate de terminalul mobil? Notaţi rezultatele obținute.
4. Care este secvența de comenzi pentru inițierea unui apel vocal la numărul 0264401571?
5. Care sunt parametrii de calitate ai semnalului recepționat? Notați rezultatele obținute.
6. Utilizați secvența de comenzi corespunzătoare pentru a afișa informații de localizare obținute de le echipamentul de tip MT. Notați rezultatele obținute.
7. Care sunt numerele stocate de telefon stocate pe cartela SIM?
8. Inițiati secvența de comenzi necesară pentru stabilirea unei conexiuni GPRS. Care este adresa IP alocată?

2.11 Serviciul de mesaje scurte (SMS- Short Message Service)

2.11.1 Introducere

Serviciul de mesaje scurte (SMS) este unul dintre teleserviciile oferite de un sistem GSM. Serviciul SMS a fost introdus incepand de la faza 1 a procesului de standardizare şi a evoluat continuu în timpul fazelor ulterioare.

2.11.2 Serviciul SMS

Specificaţiile GSM definesc serviciul SMS ca un set de resurse care permite transferul de mesaje scurte între o stație mobile GSM şi o entitate adresabilă care poate recepționa SMS-uri (*SME- Short Message Entity*), prin intermediul unui centru de servicii (*SMS-SC-SMS- Service Centre*). O entitate de tip SME poate fi un alt terminal mobil, un server sau poate fi localizată într-o rețea fixă.

În format text serviciul SMS permite transmiterea de mesaje alfanumerice de lungime maximă 160 caractere, mapate pe 140 octeți folosind alfabetul GSM implicit. Transmiterea de mesaje scurte se bazează pe conceptul ***store and forward***. Mesajele nu sunt trimise direct de la un utilizator la un altul ci sunt memorate temporar în centrul de servicii SC şi sunt transmise ulterior către destinaţia finală în momentul în care destinataruleste ataşat la reţea. Aceasta este una dintre diferențele majore între SMS și sistemele de paging folosite înainte de introducerea SMS; acestea necesitau ca destinatarii să fie atașați la rețea pentru primirea mesajelor de paging.

Memorarea în SC se poate face pentru o anumită perioadă de timp (săptămâni/luni).

Arhitectura necesară pentru serviciul SMS este ilustrată în Fig. 2.

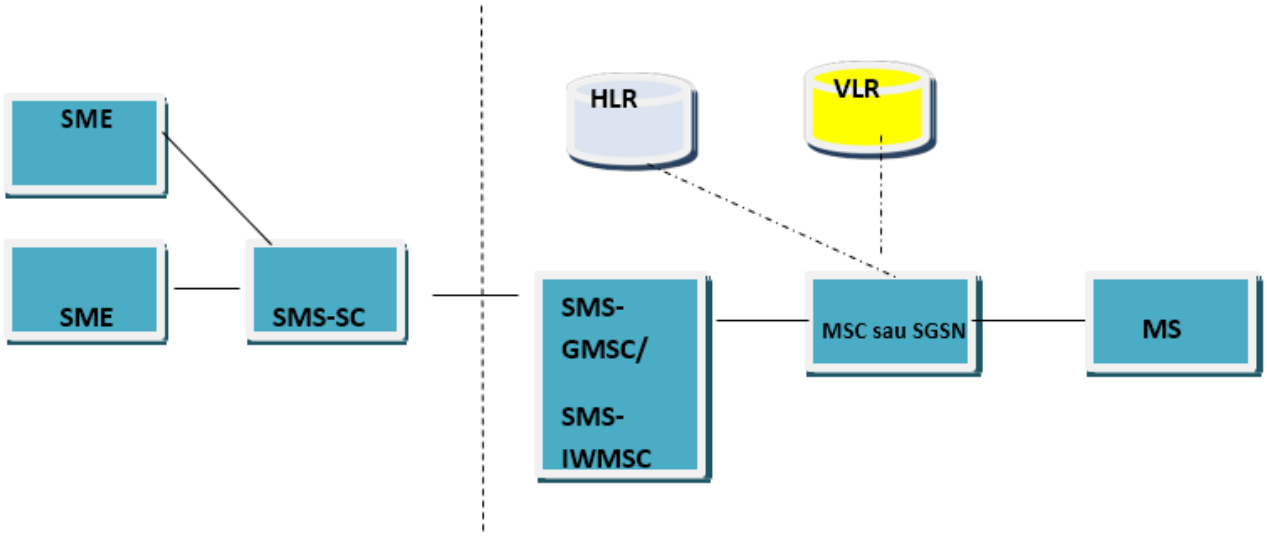


Fig.2 Arhitectură necesară pentru serviciul SMS

SME – Short Message Entity– entitate capabilă să trimită sau să recepționeze mesaje scurte (servere, terminale mobile etc)

SMS-SC – SMS Service Centre - server dedicat care permite memorarea și transmiterea SMS-urilor. Un echipament de acest tip are alocat un număr de telefon în format E.164 din gama de numere alocate unui operator.

SMS-GMSC – SMS Gateway MSC – retransmite mesajele către echipamentul MSC care deservește curent o stație mobilă. Informațiile de rutare se obține prin interogări HLR interrogation. Înaintează de asemenea rapoarte de livrare către HLR.

SMS-IWMSC – echipament care prezintă funcții necesare recepționării unui SMS și retransmiterii acestuia către SC.

HLR – furnizează informaţii de rutare pentru serviciile SMS (adresa MSC-ului curent). De asemenea, informează SC dacă utilizatorul este atașat sau nu la rețea.

În GSM transportul mesajelor de pe interfata radio este realizat prin intermediul canalelor de semnalizare dedicate (SDCCH/SACCH). Alegerea permite transmiterea simultană de SMS-uri ăi efectuarea concomitentă de apeluri vocale sau de transmisii de date.

Serviciul SMS oferă de asemenea posibilitatea confirmării livrării mesajelor prin rapoarte de livrare. Folosind această facilitate un utilizator poate fi notificat cu privire la livrarea mesajului transmis anterior la un SC. Această facilitate se activeaz de utilizatorul care trimite mesajul prin intermediul unui flag setat în cadrul corpului mesajului.

Concatenarea SMS-urilor este o altă caracteristică inclusă în specificațiile GSM. Aceasta permite transmiterea de mesaje scurte cu o lungime mai mare de 160 de caractere,

Conceptual, există două tipuri diferite de servicii de tip SMS:

-**serviciul punct la punct (*point- to- point service)*** – serviciu dedicat între doi utilizatori.

-**serviciul de difuzare în celulă (*CBS -cell broadcast service)*** – permite trimiterea de informaţii către toţi utilizatorii plasați într-o celulă

Formatul mesajelor şi modalitatea de livrare depind de tipul de serviciu folosit.

Serviciul punct-la-punct este de două tipuri:

- *Short Message Service Mobile Terminated (SMS-MT)* – definește setul de resurse folosite pentru livrarea unui SMS de la un echipament de tip SC către o stație mobilă, cu furnizarea de rapoarte de livrare şi asigurarea unor mecanisme specifice pentru livrarea ulterioară în caz de eşec.

-Short Message Service Mobile Originated (SMS-MO) **-**definește setul de resurse folosite pentru livrarea unui SMS de la un o stație mobilă către o entitate de tip SC, cu furnizarea de rapoarte de livrare şi asigurarea unor mecanisme specifice pentru livrarea ulterioară în caz de eşec.

Formatul mesajelor pentru cele două tipuri de mesaje *SMS-MT* și *SMS-MO* este diferit (SMS_DELIVER și SMS_SUBMIT în specificații [3]).

Trimiterea și recepționarea mesajelor scurte se poate face de pe terminalul mobil sau folosind comenzi AT inițiate de un echipament de tip TE pe o interfață serială. Pentru schimbul de SMS-uri via un echipament de tip MT există două posibilități diferite:

-**mod text** (neacceptat pe majoritatea terminalelor)

-mod PDU (Protocol Description Unit)

În mod text mesajele transferate prin interfaţa serială sunt împachetate în octeți folosind alfabete predefinite.

În mod PDU transferul între echipamentele de tip TE și MT se face prin conversia datelor în binar, maparea acestora pe octeți și reprezentarea hexazecimală a secvenței rezultate. Codarea şi decodarea fluxului binar se face de straturile superioare din modelul OSI. Modul de transfer PDU permite implementarea unor scheme de codare specifice iar utilizatorii au posibilitatea de a utiliza un alfabet cu un număr mai mic de biți/caracter (n) decât 7 (numărul de biţi/caracter al alfabetului GSM implicit).

Utilizând un număr de x biţi/caracter numărul maxim de caractere/SMS va creşte de la 160 la 160 * 7/x. De exemplu, utilizând un cod BCD (n = 6 biţi/caracter), un SMS va conţine 186 caractere.

Fig.3 Rapoarte de livrare /rapoarte de stare SM

2.11.3 Alfabetul implicit GSM

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	**b3**	**b2**	**b1**		0	1	2	3	4	5	6	7
0	0	0	0	0	@	Δ	space	0	|	P	¿	p
0	0	0	1	1	£	1)	!	1	A	Q	a	q
0	0	1	0	2	$	Φ	“	2	B	R	b	r
0	0	1	1	3	¥	Γ	#	3	C	S	c	s
0	1	0	0	4	è	Λ	¤	4	D	T	d	t
0	1	0	1	5	é	Ω	%	5	E	U	e	u
0	1	1	0	6	ù	Π	&	6	F	V	f	v
0	1	1	1	7	ì	Ψ	„	7	G	W	g	w
1	0	0	0	8	ò	Σ	(	8	H	X	h	x
1	0	0	1	9	Æ	Θ	)	9	I	Y	i	y
1	0	1	0	10	LF	Ξ	*	:	J	Z	j	z
1	0	1	1	11	Ø	1)	+	;	K	Ä	k	ä
1	1	0	0	12	ø	Æ	,	<	L	Ö	l	ö
1	1	0	1	13	CR	æ	-	=	M		m	ñ
1	1	1	0	14	Å	ß	.	>	N	Ü	n	ü
1	1	1	1	15	å	Ç	/	?	O	§	o	à

Tabelul 1. Alfabetul GSM pentru SMS-uri

2.11.4 Comenzi AT pentru SMS-uri

Trimiterea și recepționarea de SMS-uri pe un echipament distant este permisă prin intermediul comenzilor AT. Acestea sunt indicate în specificația GSM 07.05 [2]. Cele mai importante comenzi sunt rezumate îm tabelul de mai jos.

Comandă	Descriere	Răspuns	Observații
AT+ CSCA=<sca>,[<tosca>]	Setare număr SC pentru serviciul SMS-MO	OK	**<sca>** – MSIDSN pentru SC **<tosca>** - tip număr penru SC 161- național ,145 internațional Ex: AT+CSCA =”+40722004000” Comanda poate fi ignorata daca adresa SC stocată pe cartela SIM este utilizata. Adresa curentă a SC poate fi determinată folosind comanda AT+ CSCA? <sca> poate fi trimisă debutul unui mesaj scurt în modul PDU
AT + CMGF=mode	Setare mod transfer	OK	Dacă **mode** =0 va fi folosit modul PDU iar dacă **mode**=1 va fi setat modul text. Comanda de test asociată este AT+CMGF=? command
AT + CSMS=<service>	Selectare tip SMS (CBS, SMS-MO, SMS-MT)	+CSMS: mt, mo, bm OK	**service** =0 comenzi AT faza 2 de standardizare **service =1** comenzi AT faza 2+ destandardizare **mt** – 0 MT neimplementat **mt** – 1 MT implementat **mo** – 0 MO neimplementat **mo**– 1 MO implementat **bm** – 0 mesaje de difuzare neimplementate **bm** – 1 mesaje de difuzare implementate
AT+CPMS=<mem1>, <mem2>, <mem3>	Selectarea locației de memorie pentru citire/ștergere <mem1> scriere/trimitere <mem2> recepție <mem3>	+CPMS: <used1>, <total1>, <used2>, <total2>, <used3>, <total3> OK	Exemplu : <mem1> =<mem2> =<mem3> =“SM” este o opțiune care selectează cartela SIM. Valoarea “ME” setează memoria terminalului
AT + CGMR=<index>	Citirea unui mesaj de la adresa <index > selectata prin comanda AT +CPMS	In mod PDU: +GMGR:<stat>,<length><CR><LF> <pdu>><CR><LF> OK	<stat> - 0- mesaj necitit, 1 mesaj citit, 2 – mesaj necitit memorat, 3-mesaj trimis memorat <length> - mărimea mesajului în octeți (doar pentru mod PDU) <pdu> mesajul în format PDU AT + CGMR=? – comandă de test

Comandă	Descriere	Răspuns	Observații
AT+ CMSS=<index>[,<da> [,<toda>]]	Trimitere mesaj din memoria selectată prin comanda AT+CPMS command	+CMSS: <mr> <mr> (valoare care se autoindexează) OK	<index> - index către locația mesajului <da> - numărul de telefon al destinatarului <toda> tip număr, national, internațional
AT+CMGS= <length> <CR> *PDU message* <ctrl-Z / ESC >	Trimitere mesaj în format PDU	+CMGS: <mr> <mr> (valoare care se autoindexează) OK	<length > - mărimea în octeți a mesajului PDU- și în reprezentare hexazecimală Ctrl-Z caracter final ESC pentru anulare
AT+CMGW= <length> [,<stat>] <CR> *PDU message* <ctrl-z/ESC>	Scriere mesaj în memorie	+CMGW: <index> OK	
AT +CMGD=<index>	Ștergere mesaj	OK	<index> locația mesajului
AT + CMGL=<stat>	Listare mesaje din memorie cu starea indicate de parametrul <stat>	+CMGL: <index>,<stat>, ,<length>,<pdu><CR><LF>	<stat> - 0 mesaje necitite 1- mesaje citite memorate 2- mesaje necitite memorate 3- mesaje trimise memorate 4- toate mesajele
AT +CSAS=<profile>	Salvare setări	OK	

2.11.5 Formatul mesajelor asociate mesajelor de tip SMS-MT și SMS-MO (mod PDU)

Formatul mesajelor este ilustrat în Fig.4 (SMS-MT) și 5 (SMS-MO):

SCA	PDU type	OA	PID	DCS	SCTS	UDL	UD
1-10 octeți	1 octet	2-12 octeți	1 octet	1 octet	7 octeți	1 octet	0-140 octeți

Fig.4 Format SMS-MT. *Abrevieri: SCA- Service Centre Address; PDU-Protocol Data Unit; OA-Originator Address ; PID-Protocol Identifier; DCS-Data Coding scheme ; SCTS- Service centre time stamp ; UDL-;*

1-10 Octeți	1 octet	1 octet	2-12 octeți	1 octet	1 octet	1/7 Octeți	1 octet	0-140 octeți
SCA	**PDU type**	**MR**	**DA**	**PID**	**DCS**	**VP**	**UDL**	**UD**

Fig.5 Format SMS-MO. *Abrevieri:SCA- Service Centre Address; PDU-Protocol Data Unit;DA-Destination Address ; PID-Protocol Identifier; DCS-Data Coding scheme ; VP- Validity Period; UDL- user data length ; UD- user data*

2.11.6 Codarea parametrilor

- **SCSA –***Service Centre Address* – numărul de telefon asociat SC. Formatul utilizat este următorul:

length	number type	BCD digits
1 octet	1 octet	0-8 octets

lenght- numarul de octeți pe care se reprezintă numărul de telefon al SC plus un octet (tipul acestuia). Valoarea tipică este 7. Dacă câmpul are valoarea "00" numărul de telefon al SC de pe cartela SIM este utilizat iar celelalte câmpuri sunt ignorate.

number type - 129 = 81H (national) și 145 = 91H (international)

BCD digits – utilizate pentru reprezentarea numărului de telefon al SC; un octet include 2 cifre BCD inversate (format *swapped nibble)***:**

octet 1	octet 2	…	Ultimul octet
digit2 | digit1	**digit 4|digit 3**	**…**	**Ultimul digit/sau FH** **Ultimul digit/sau 0**

Exemplu: +40744946000 *(SC Orange Romania) se reprezintă prin* ***07 91 0447946400F0***

- **PDU** – *Protocol Data Unit Type*

Identifică tipul de mesaj (SMS-MO sau SMS-MT). Permite activarea rapoartelor de livrare a mesajelor, setarea modului de reprezentare a perioadei de valabilitate etc. O valoare tipică pentru mesaje de tip MT este 04H în timp ce pentru mesaje de tip MO aceasta este 11H.

- **OA/DA – Originator/Destination Address**

Ambele numere de telefon au același format, fiind codate similar câmpului SCSA, diferența constând în modul de reprezentare a lungimii, aceata reprezentând lungimea fizică a numerelor

Exemple: + 40744991518 *se reprezintă prin* ***0B 91 0447941915F8.*** *Același număr în format national 0744991518 are reprezentarea asociată* ***OA 81 7044775181***

- **MR – Message reference**

Octet folosit doar pentru mesaje de tip MO pentru a permite indexarea la nivel SC a mesajelor trimise de aceeași entitate. O valoare "00H" permite MT să autoindexeze mesajele transmise.

- **PID – Protocol identifier**

Octetul este utilizat pentru a transmite către SC care sunt protocoalele de strat superior care trebuie folosite pentru livrarea mesajelor. Inițiatorul mesajului poate a seta acest câmp. Valoarea tipică pentru SMS-uri este 00H.

- **DCS – Data coding scheme**

Câmpul permite indicarea alfabetului folosit și a clasei SMS-ului. Modul de reprezentare este indicat mai jos:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	0	b5	b4	b3	b2	b1	b0

b5 = **0/1** – datele utilizator sunt / nu sunt comprimate
b4 =**0/1** – b1 și b0 nu sunt folosiți/ b1 și b0 indică clasa
b3 b2 – informații legate de alfabet: **00** –alfabet GSM pe 7 biți, **01** - 8 biti (ASCII extins), **10** – Unicode (16 bit)
b1b0 – 00 – mesaj de clasă 0 (afișare imediată), alte valori- mesaje specifice TE sau ME.
Exemple: DCS=00H- alfabet GSM, fără compresie, mesaj de clasă 0; **DCS=04H –** codare pe 8 biți

- **VP – validity period**

Câmp optional care indică perioada după care mesajul poate fi şters din SC dacă nu a fost deja livrat. Informaţiile sunt reprezentate pe un octet sau pe 7 octeți.

În cazul unei reprezentări pe un octet (cea mai des utilizată) exprimarea validității este relativă, cu începere de la momentul la care SC a primit mesajul. Modul de calcul pentru o reprezentare de tipul b7 b6 b5 b4 b3 b2 b1 b0 presupune conversia în zecimal (N). Valoarea se determină mai apoi astfel:

- dacă N este în intervalul 0-143 - VP=(N+1)*5 minute;
- dacă N este în intervalul 144-167-VP=12h+(N-143)*30 minute;
- dacă N este în intervalul 168 -196 -VP=(N-166) *1 zi
- dacă N este în intervalul 197 – 255 – VP= (N-192)*1 săptămână

Exemplu: VP de 5 zile se reprezintă prin ABH

În reprezentare absolută perioada de valabilitate este reprezentată pe 7 octeţi într-un format de tipul an/lună/zi/oră/minut/Second/fus orar, similar cu formatul de SCTS

- **SCTS – Service Centre Time Stamp**

. Câmpul este utilizat pentru a transfera informații de la SC către SME/MS legat de momentul la care a fost recepționat mesajul. Formatul folosit este

Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7
Y2 | Y1	**Mo2 | Mo1**	**D2 |D1**	**H2|H1**	**Mi2 | Mi1**	**S2|S1**	**TZ2 |TZ1**

Exemplu: *Codificarea SCTS 50 01 72 01 03 00 80* indică că mesajul a fost primit în 27 octombrie *2005 la ora 10h30min00sec GMT+2h (România TZ1TZ2x15 min=2h)*

- **UDL – user data length**

  Reprezintă lungimea câmpului UD. O valoare de 0AH indică un câmp UD de 10 caractere.

- **UD – user data**

Codifică conţinutul actual al mesajului utilizând alfabetul specificat de câmpul DCS. Cel mai utilizat alfabet este cel implicit cu reprezentare pe 7 biți. Reprezentarea pe octeți resupune prelucrarea șirului binar rezultat folosind procedura explicată mai jos.

Fie b_{11} b_{12} b_{13} b_{14} b_{15} b_{16} b_{17} reprezentarea pe 7 biți a primului caracter, b_{21} b_{22} b_{23} b_{24} b_{25} b_{26} b_{27} a celui de-al doilea, b_{31} b_{32} b_{33} b_{34} b_{35} b_{36} b_{37} a celui de-al treilea s.a.m.d. cu bitul al 7 -lea cel mai semnificativ. Figura 8 ilustrează modul de reprezentare a fluxului initial în reprezentare pe 8 biți.

b_{21}	b_{17}	b_{16}	b_{15}	b_{14}	b_{13}	b_{12}	b_{11}
b_{32}	b_{31}	b_{27}	b_{26}	b_{25}	b_{24}	b_{23}	b_{22}
b_{43}	b_{42}	b_{41}	b_{37}	b_{36}	b_{35}	b_{34}	b_{33}
b_{54}	b_{53}	b_{52}	b_{51}	b_{47}	b_{46}	b_{45}	b_{44}
b_{65}	b_{64}	b_{63}	b_{62}	b_{61}	b_{57}	b_{56}	b_{55}
b_{76}	b_{75}	b_{74}	b_{73}	b_{72}	b_{71}	b_{67}	b_{66}
b_{87}	b_{86}	b_{85}	b_{84}	b_{83}	b_{82}	b_{81}	b_{77}
0	b_{97}	b_{96}	b_{95}	b_{94}	b_{93}	b_{92}	b_{91}

Fig.4 Împachetarea datelor pe 7 biți într-un format pe octeți

Exemplu: Utilizând alfabetul GSM pe 7 biți mesajul – " TEST 1"- va fi codat după cum urmează.

TEST 1 --> 1010100 1000101101001 1010100 0010000 0011000

1	1	0	1	0	1	0	0
1	1	1	0	0	0	1	0
1	0	0	1	0	1	0	0
0	0	0	0	1	0	1	0
1	0	0	0	1	0	1	0
0	0	0	0	0	0	0	1

Citirea mesajului rezultat se face pe linii, reprezentările în hexazecimal rezultate fiind concatenate.

D4E2940A8A01

Întrebări

1. Care este numărul maxim de caractere care pot fi trimise folosind SMS?
2. Ce avantaje oferă conceptul store-and-forward asociat serviciului SMS?
3. În cazul în care un utilizator este implicat într-un apel vocal sau sesiune de date va putea primi/trimite SMS-ul? Argumentați răspunsul.
4. Când un utilizator trimite un mesaj scurt, acesta utilizează formatul SMS-MT sau SMS-MO?
5. Care sunt avantajele modalității de de transmitere în format PDU față de varianta text?
6. Ce avantaje oferă folosirea comenzilor AT pentru SMS-uri?
9. Care sunt comenzile utilizate pentu citirea/trimitere de SMS-uri?
10. Cum poate fi configurat modul PDU de transmitere la nivel MT?
11. Care sunt diferenţele principale dintre formatele SMS-MO şi SMS-MT? Descrieţi funcţionalitatea câmpurilor diferite.
12. Codați în formatul corect urmatoarele valori: adresa centrului de servicii: + 40722006000, VP: 4 zile, UD: SIM

2.11.7 Exerciții

1. Alegeti un text arbitrar pentru corpul unui SMS. Codificați-l manual și verificați rezultatul folosind convertorul online de la adresa http://rednaxela.net/pdu.php.
2. Care este lungimea şirului hexazecimal? Care ar fi lungimea secvenţei în cazul în care ar fi folosit un alfabet de 8 biţi?
3. Care este secvenţa de la comenzi necesare pentru a trimite un SMS în modul PDU folosind dispozitivele de tip furnizate la laborator?
4. Utilizaţi aplicaţia Tera Term pentru a trimite SMS în modul PDU “0011000B91xxxxxxxxxxx0000AA06D4E294FA8F01" la un alt număr din cele folosite la laborator (Indicație secvența "xx...xxx" trebuie să fie reprezentarea unui număr valid E. 164 asociat unei cartele SIM utilizate la laborator.
5. Care este secvenţa de la comenzi care pot fi utilizate pentru a lista toate mesajele stocate pe cartela SIM?
   Dar dacă locaţia de stocare ar echipamentul mobil?
6. Pe baza răspunsului la întrebarea anterioară, aflați care SMS-urile stocate pe cartela SIM. Conectați-vă la MT și afișati în consolă unul dintre aceste mesaje

Bibliografie

[1]. 3GPP TS 07.07 AT command set for GSM Mobile Equipment (ME) (Release 1998 (link)
[2]. S. Redl, N. Weber, M. Oliphant – GSM and Personal Communications Handbook, Artech House Publishers, 1998
[3]. GSM 03.40 –Technical realization of the Short Message Service (link)
[4]. GSM 07.05 - Use of Data Terminal Equipment - Data Circuit terminating;Equipment (DTE - DCE) interface for Short Message Service (SMS) and Cell Broadcast Service (CBS) (link)
[5]. GSM 03.38 –Alphabets and language specific information (link)
[6]. ETSI TS 127 007 - Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); AT command set for User Equipment (UE) , version 8.3.0 Release 8.
.

Lucrarea 3 Aplicații web mobile

3.1 Introducere

Aplicațiile web mobile sunt aplicații care rulează pe un browser mobil. Acestea sunt de fapt pagini web care necesită conectivitate internet și au aspectul și funcționalitatea aplicațiilor native, proiectate și implementate folosind tehnologii specifice pentru smartphone-uri cu un anumit sistem de operare (Android, IOS sau WM) sau folosind o anumită tehnologie (JME-Java pentru dispozitive embedded).

Standardul WAP (Wireless Application Protocol) a fost primul standard care a permis utilizarea browser-elor terminale mobile pentru a accesa informații de pe internet și pentru a proiecta și rula aplicații care utilizează tehnologia internet.

Inițial, telefoanele mobile şi internetul erau două tehnologii care nu aveau nimic în comun; odată cu apariţia WAP, informațiile de pe internet au putut fi accesate și de utilizatori mobili. Introducerea WAP a adus beneficii atât pentru utilizatorii mobili, cât și pentru furnizorii de servicii. Utilizatorii mobili au putut utiliza un unic dispozitiv pentru servicii de telefonie și servicii de date similare cu cele pe care le accesează de la un PC/laptop conectat la internet: ştiri, rezultate sportive, informații bursiere, etc. Pentru operatorii de telefonie mobilă WAP a permis furnizarea de noi servicii, independente de locația utilizatorului sau, dimpotrivă, dezvoltarea de noi, servicii bazate pe localizare.

Istoria WAP este strâns legată de cea a **WAP forum** (1997), un grup de lucru compus din producători de terminale mobile de tip *feature phone*, operatori de reţea şi dezvoltatori de aplicaţii. În interiorul acestui grup un rol major a fost jucat de către compania de software OpenWave care a fost producătoarea principală de browsere pentru telefoanele mobile.

Specificaţiile WAP emise au fost structurate pe versiuni (WAP 1.1-propus în 1999, WAP 1.2-2000 şi WAP 2.0-2001). În prezent dispozitivele mobile din generațiile 2G și 2.5 G sunt compatibile WAP 1.1 şi WAP 1.2, în timp ce pentru terminale mobile 3G, WAP 2.0 este standardul de facto. Din 2002, forumul WAP a fost integrat în Open Mobile Alliance, un organism internaţional de standardizare care oferă specificaţii atât pentru WAP cât şi pentru alte tehnologii wireless dedicate furnizării de servicii de date mobile.

Printre tehnologiile concurente, *I-mode* a fost folosit pentru acces la Internet mobil în Japonia şi Asia. În zilele noastre, odată cu utilizarea pe scară largă a terminalelor de tip smartphone, tehnologiile web mobile au evoluat de la xHTML-MP şi WAP-CSS (WAP 2.0) la HTML5 și Javascript. Modelul de comunicare WAP este încă utilizat pentru furnizarea de servicii pentru terminale de tip *feature phone*, pentru serviciul MMS şi pentru notificările de tip WAP Push.

3.2 Modelul de conectivitate WAP

Pentru furnizarea de servicii de internet, standardul WAP folosește arhitectura din figura 1.

Aplicaţiile WAP şi conţinutul WAP au fost dezvoltate folosind un format similar cu cel utilizat în **World Wide Web (WWW)** pentru a permite schimbul și transferul de diferite tipuri de documente. **WAE (Wireless Application Environment)** specifică în WAP toate elementele necesare pentru dezvoltarea de aplicații: limbaj de marcare (eng. *markup*), **WML (Wireless Markup Language)**, script-uri pe partea de client (**WML script**). Script-urile WML şi WML sunt utilizate pentru a descrie conținutul WAP (imagini, pagini dinamice sau statice) într-un mod similar celui în care HTML este folosit pentru a descrie conținutul paginilor clasice.

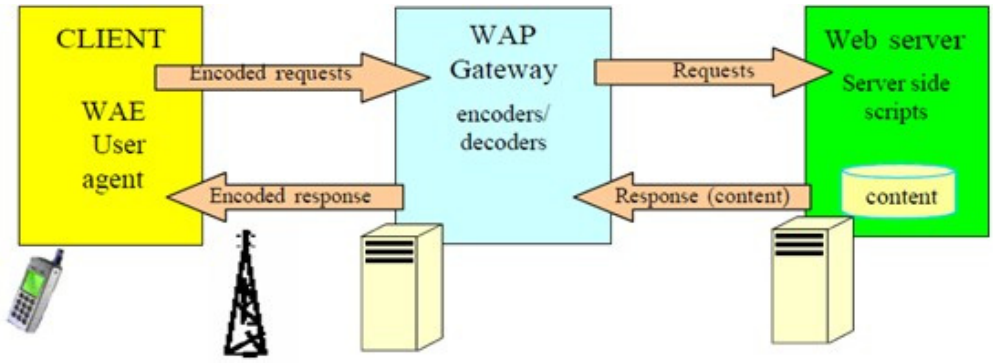


Fig.1 Model de conectivitate WAP

Identificarea documentelor pe internet de face clasic folosind identificatori de tip **URL (Uniform Resource Locator)**, de tipul **protocol://hostname/filename.** Accesarea unui document prin intermediul WAP presupune următoarele etape:

-utilizatorul selectează URL-ul asociat informațiilor dorite;

-browser-ul telefonului mobil transmite cererea URL către entitatea Gateway WAP folosind un format binar, utilizând tehnici de compresie bazate pe token-uri;

-Gateway-ul WAP convertește cererea în format web clasic, de exemplu folosind **HTTP (Hypertext Transfer Protocol)** şi o plasează pe Internet;

- serverele de web procesează informația în mod similar solicitărilor de tip HTTP (accesează documentul structurat WML cu elemente/etichete (eng. *tag*) adăugând sau nu conținut dinamic și îl returnează adăugând informații legate de modul în care trebuie afișat documentul prin intermediul mecanismului **MIME (Multipurpose Internet Mail Extensions)** – *text/html, image/png* etc.). Un antet HTTP standard este adăugat în mesajul de răspuns către gateway-ul WAP.

-acesta elimină antetul HTTP şi codifică datele într-un format binar care va fi transmis pe telefonul mobil;

- micro browser-ul de pe terminalul mobil decodează conținutul WML şi îl afişează.

Standardul WAP a fost definit folosind o arhitectură stratificată de tip OSI care permite dezvoltarea flexibilă de aplicații, orientate pe conexiune sau nu, cu folosirea de facilități de compresie/decompresie, confirmare locală a pachetelor etc. Stratul cel mai important este WAE iar acesta definește:

-un limbaj de marcare pentru descrierea conținutului paginilor web de pe browsere compatibile WAP **WML/XHTML-MP (Extensible Hypertext Markup Language – Mobile Profile)**;

- facilități de implementare de logică procedurală în aplicația client prin folosirea de script-uri pe partea de client -**WML script**;

-metode (eng. *API*) specifice care permit efectuarea de apeluri telefonice direct din paginile web;

-suport MMS;

Wireless Markup Language (WML)

În mod similar cu modalitatea de descriere a paginilor folosind limbajul de marcare HMTL clasic, WML este utilizat în WAP pentru a specifica aspectul paginilor, pentru a controla navigarea într-un site

WAP, pentru a permite accesul la paginile din afara site-ului WAP și pentru a furniza facilități de introducere a datelor.

WML a fost derivat din XML (Extensible Markup Language) şi utilizează unele concepte specifice pentru a organiza datele care urmează să fie afişate de către browserul mobil:

-**card** -o pagină care urmează să fie afișată este împărțită în sub pagini; o astfel de sub pagină se numeşte card în terminologie WAP. Un singur card este afișat la un moment dat.

-**deck** – este o colecție de elemente de tip card; navigarea între card-uri din același deck este locală, fără a implica un dialog suplimentar client-server.

WML suportă o mare varietate de dispozitive (telefoane mobile de tip feature phone, PDA-uri etc). Cele mai multe medii de dezvoltare permit utilizarea unui simulator pentru dezvoltarea de aplicații, cu emularea funcționalității terminalelor de tip feature phone: taste pentru navigare înainte și înapoi, taste programabile (*soft keys*), taste direcționale pentru navigarea în pagină. Există de asemenea aplicații care permit simularea comportamentului unei aplicații pe acest tip de terminale (de exemplu Opera Mobile Emulator) (fig.2)

Fig.2 Simulatoare WAP

WML, similar HTML, este un limbaj de marcare care descrie conținutul unei pagini prin etichete. O pagină începe cu eticheta **<wml>** şi se termină cu **</wml>**. Un document WML trebuie să includă un antet specific care permite identificarea tipului de limbaj de marcare folosit, setul de reguli pentru limbajul de marcare (**DTD – Document Type Definition).**

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

3.3 WMLScript

WML script a fost introdus cu scopul de a îmbogăţi funcţionalitatea WML, pentru a permite implementarea de logică procedurală în aplicațiile WML (funcții, variabile, bucle, condiții etc), facilitarea accesului la funcții de telefonie și SMS din interiorul unei aplicații care rulează în browser precum și pentru a permite dezvoltării de pagini dinamice.

Codul WML Script este compilat pe gateway-ul WAP într-un format binar. Dispozitivul mobil şi gateway-ul WAP transferă date în acest format binar. Avantajul unei astfel de abordări este că nu este nevoie de un compilator pe dispozitivul mobil, permițând astfel o utilizare mai eficientă a resurselor terminalului mobil.

3.4 XHTML-MP

Reprezintă un limbaj de marcare adoptat în specificațiile WAP 2.0. Adaugă funcționalități noi față de WML, unificând modul de dezvoltare a aplicațiilor web mobile cu cel folosind pentru aplicațiile clasice, adăugând suport CSS pentru formatarea conținutului. În prezent, marea majoritate a paginilor WAP sunt dezvoltate folosind această tehnologie.

3.5 Tehnologia WAP push

În contrast cu tehnologia "pull" utilizată în comunicațiile WAP clasice (Fig. 1), tehnologia PUSH introdusă pentru prima dată de standardul WAP, permite afișarea de date în browser-ele terminalelor mobile fără o cerere de tip GET prealabilă.

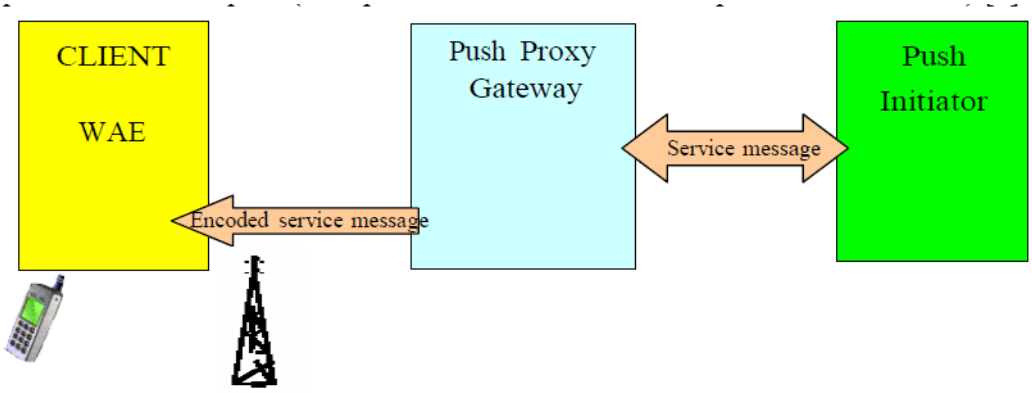


Fig.3 Model de conectivitate WAP PUSH

Entitatea funcțională ***Push Initiator*** este o aplicație server care inițiază transmiterea unui mesaj de tip WAP push. Transmiterea propriu-zisă a mesajului este realizată prin intermediul unui echipament de tip ***Push Proxy Gateway (PPG)***, care formatează mesajul într-un format specific pentru a-l transmite către mobil, folosind drept servicii suport GPRS sau serviciul SMS. Mesajul formatat include un număr de port care permite terminalului mobil să interpreteze și să afișeze conținutul mesajului în browser, nu în aplicația de mesagerie. Standardul WAP definește mai multe mesaje de tip WAP push. Cele mai importante sunt:

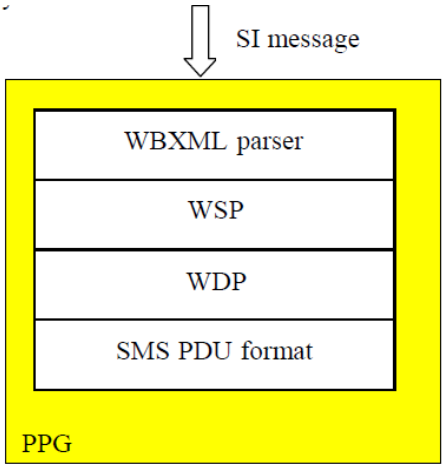
-***Service Indication*** **(SI)**- mesajele care includ următoarele tipuri de informații: informații de tip text (notificări) care a urmează a fi afișate de browser și, opțional, link-uri pentru informații suplimentare ce pot fi accesate direct din browser.

-***Service Loading*** (SL)- nu conțin informații text pentru utilizatorul mobil ci inițiază acțiuni directe din browser, fără intervenția utilizatorului.

3.5.1 Exemplu de codare a unui mesaj de tip WAP push de tip SI pentru transmitere prin SMS

Procesul de codare va fi ilustrat pe baza unui mesajului de mai jos:

<?xml version="1.0"?> <!DOCTYPE si PUBLIC "-//WAPFORUM//DTD SI 1.0//EN" "http://www.wapforum.org/DTD/si .dtd">	DTD
<si>	Mesaj de tip SI cu afișare imediată
<indication	
href=http://wap.google.com si-	Link pentru browser
id=5> action="signal-high"	
mcs 2006 </indication> </si>	notificare



Ieșirea blocului de analiză **WBXML parser** este:

WSP (Wireless Session protocol) este un protocol WAP dedicat care adaugă o serie de funcționalități pentru mesajele de tip WAP Push:

-referențierea mesajelor multiple de tip push

-identificarea tipului de conținut MIME şi lungimea pentru tipul de conținut

Pentru exemplul de mai sus, protocolul WSP adaugă următoarele informații în antet:

7A0601AE

7A- Push ID (valoare cu autoincrementare)

06 –valoare predefinită pentru mesaje de tip PUSH

01 – conținut MIME reprezentat pe un octet

AE – token pentru tipul MIME application/vnd.wap.sic.

WDP (Wireless Datagram Protocol) este un protocol WAP din stratul transport ce permite stabilirea unei conexiuni cu browser-ul terminalului mobil prin intermediul unor porturi predefinite. Pentru exemplul de mai sus informațiile adăugate de protocolul WDP sunt:

0605040B8423F0

06- lungime header WDP în octeți

05- lungime a informației legată de porturi (1 octet pentru lungime + 2 octeți pentru porturile destinație și sursă)

0B84 – valoare predefinită port destinație (browser compatibil WAP)

23F0 – valoare port sursă (orice valoare este posibilă, comunicația fiind într-un singur sens.

În final mesajul este împachetat și transmis în câmpul UD dintr-un SMS. Antetul mesajului de tip SMS-MO include informațiile prezentate în lucrarea de laborator anterioară. În ipoteza în care MSISDN-ul destinatarului ar fi +40744991816, antetul adăugat are expresia:

0061000C910447941918F600F5

00 – adresă SC de pe cartela SIM

61 – mesaj de tip SMS-MO cu confirmare

00 – index (orice valoare)

0C – lungimea numărului destinatarului (în digiți)

91 – format internațional

0447941918F6 – format swapped nible pentru numărul destinatarului

00 – PID

F5 – DCS – 8 biți

Câmpul UDL codează în hexazecimal numărul de octeți al câmpului UD. Pentru mesajul anterior formatul SMS-ului rezultat (PUSH PDU) este:

0061000C910447941918F600F5300605040B8423F07A0601AE01056A0045C60C037761702E676F6F676C652E636F6D000801036D63732032303036000101.

Trimiterea mesajului folosind comenzi AT se face cu : AT + CMGS = 61 <PUSH PDU><ctrl-z>

3.6 Aplicații web mobile folosind HTML5

3.6.1 Introducere

Odată cu introducerea XHTML-MP dezvoltarea aplicațiilor web mobile se face utilizând același model folosit pentru dezvoltarea aplicațiilor clasice.

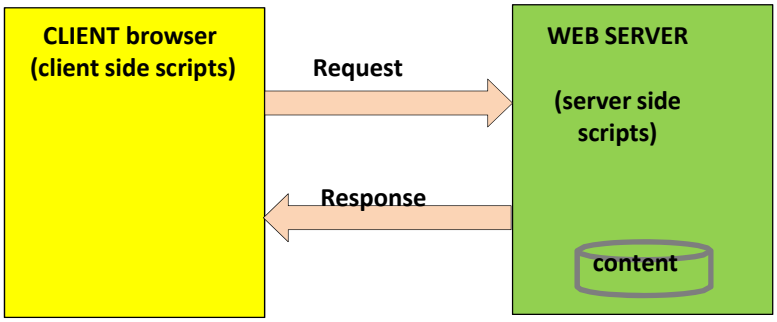


Fig.4. Model simplificat pentru dezvoltarea de aplicații web mobile

Similar WAP, pentru dezvoltarea de aplicații se utilizează script-uri, aplicații înglobate în fișierul HTML sau secvențe de instrucțiuni pe server. Acestea pot rula în browser-ul terminalului mobil (*client-side scripts*) sau pe serverul de web (*server-side scripts*). Aceste aplicații pot implementa logică procedurală sau pot modifica conținutul paginii web în mod dinamic. În esență, script-urile pe partea de client permit modificarea paginilor web după ce acestea au fost furnizate terminalului mobil în timp ce script-urile de server sunt utilizate pentru crearea sau modificarea paginii web înainte ca acestea să fie returnate terminalului mobil.

Aplicațiile web mobile utilizează uzual tehnologia **JavaScript** ca tehnologie pe partea de client în timp ce pentru servere sunt folosite tehnologii precum **ASP** sau **PHP**.

Cele mai multe telefoane mobile de sprijin în prezent HTML5 ca limbaj de marcare utilizat pentru a structura şi de a prezenta conținutul de pe browser-ul web mobil.

3.6.2 HTML5 și CSS3

HTML5 este standardul de facto în prezent pentru structurarea conținutului unei pagini web mobile. Similar WAP, HTML5 folosește etichete pentru a structura și afișa acest conținut. Structura unei aplicații simple de tip *Hello worl*d în HTML5 este ilustrată mai jos folosind simulatorul w3schools:

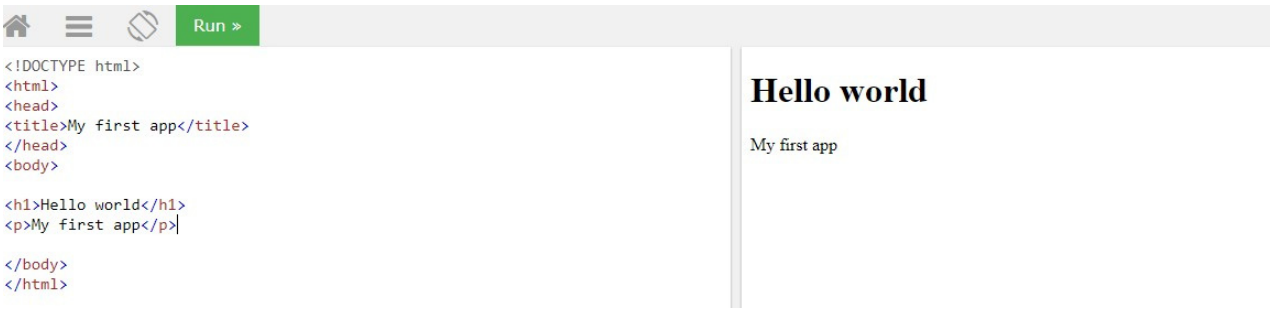


Fig. 5 Aplicația "Hello world" HMTL5 simulată pe un terminal de tip desktop

În HTML5 eticheta **<!DOCTYPE>** e mai simplă și nu referențiază un set de reguli de tip **DTD**.

Eticheta inclusă **<html>** indică browser-ului faptul că pagina este scrisă în HTML5.

Partea vizibilă a unei pagini trebuie inclusă între etichetele **<body>** și **</body>** iar titlul acestei se definește cu eticheta **<title>**. Conținutul propriu zis al unei pagini este descris prin alte etichete, în formatul impus: <**tagname>……..</tagname>** (** și ** sau **<h1> și </h1>** de exemplu).

Eticheta **<head>** este utilizată pentru a regrupa informații generale despre pagină, permițând și includerea de metadate (informații care nu se afișează dar care pot controla modul în care o a pagină este afișată de exemplu elemente tip **viewport** pentru caracteristici de tip *responsive design*, ce permit auto scalarea conținutului afișat funcție de caracteristicile mediului de afișare: rezoluție, orientare etc).

Conținutul paginii poate fi formatat folosind un limbaj dedicat care permite utilizarea de stiluri într-un document HTML5. Limbajul dedicat pentru HTML5 este *CSS3 (Cascading Style Sheets 3)*. Instrucțiunile de formatare CSS pot fi adăugate unei pagini HTML5 fie prin utilizarea unui fișier separat, fie în documentul HTML în sine, utilizând atribute ce pot fi predefinite prin eticheta **<style>** în secțiunea **<head>** a documentului. În figura 8 este ilustrat modul în care pot fi aplicate stiluri aplicației *Hello world* folosind a doua variantă.

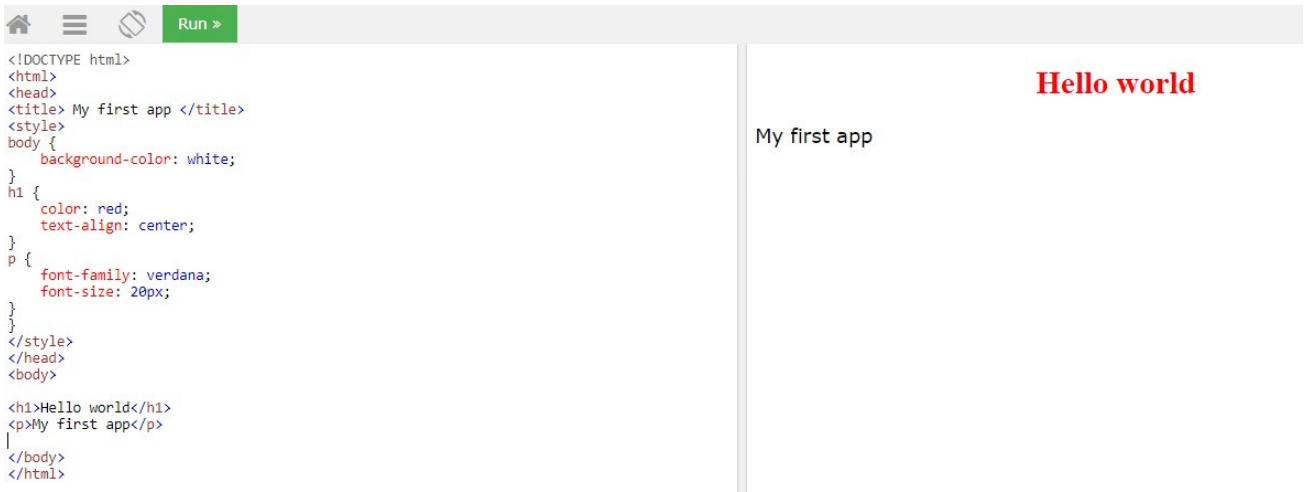


Fig.6 Aplicația "Hello world" folosind HTML5 și CSS

3.6.3 Exerciții

1) Descărcați proiectul plasat pe pagina de web a disciplinei. Arhiva include scheletul unei aplicații realizate folosind framework-ul Cordova/PhoneGap ce permite dezvoltarea de aplicații hibride în licență open source. Tehnologia Cordova/Phonegap include script-uri pe partea de client cu metode ce permit interfațarea cu partea hardware a terminalelor mobile și metode pentru conectivitate Web. Proiectul modifică metoda *OnCreate* a unei aplicații Android tipice pentru a permite afișarea de conținut HTML5 fără interacțiune cu un server web dedicat.

2) Dezarhivați proiectul și importați-l în mediul de dezvoltare Eclipse.

3) Lansați un simulator de terminal mobil de tip *Android Virtual Device (AVD)* și verificați modul în care informația este afișată.

4) Creați un nou simulator de tip AVD cu caracteristici diferite (rezoluție ecran de exemplu) și repetați punctul 3).

5) Formatați pagina folosind fișierul **index.css** inclus în proiect sau procedând de o manieră similară celei indicate în Fig. 7.

6) Stergeți conținutul fișierului **index.html**. Folosind informațiile de pe pagina https://www.w3schools.com/html/default.aspconstruiți o pagină web mobilă care să permită vizualizarea conținutul paginii http://ares.utcluj.ro/cm_2018.html.Utilizați mai multe tipuri de etichete pentru a obține același efect.

3.7 Dezvoltarea de aplicații web mobile folosind script-uri pe partea de server

Deși prin utilizarea de script-uri pe partea de client pot fi adăugate funcționalități suplimentare, aspecte precum modificarea dinamică a conținutului, minimizarea cantității de informații transferate în dialogul client-server nu pot fi obținute prin folosirea de tehnici de programare doar pe partea de client. Acestea pot fi însă implementate prin utilizarea de script-uri pe server, folosind tehnologii precum PHP, ASP sau tehnici Java de tip JSP/Servlets. Secțiunea următoare detaliază modul în care aplicațiile HTML5 se pot interfața cu script-uri ASP.

3.7.1 Active Server Pages

Microsoft Active Server Pages (ASP) este o tehnologie ce permite dezvoltarea de script-uri pe partea de server. Într-o interacțiune browser-server clasică de tip GET un astfel de script se execută înainte de returnarea datelor către browser-ul utilizatorului.

Script-urile ASP sunt declarate în fișiere cu extensia **. ASP** şi se bazează pe limbajul VBScript, derivat din Visual Basic.

Un fișier ASP trebuie să includă o linie de tipul:

```
<% @LANGUAGE="VBSCRIPT" %>
```

ce indică faptul că instrucțiunile din fișier respectă sintaxa definită de VB script și că trebuie procesate folosind acest limbaj.

Codul VBScript poate fi de asemenea inclus în pagini HTML folosind aceleași caractere de delimitare <% %>; textul care nu este cuprins între acestea va fi tratat ca html.

3.7.2 Utilizarea ASP pentru a accesa informații dintr-o bază de date

Pe lângă simplitatea sa, unul dintre punctele forte ale VBScript este capacitatea sa de a lucra cu baze de date folosind obiecte **ActiveX** predefinite:

-clase și obiecte de tip **Server** – permit accesul la metode și proprietăți pe serverul de web. Pentru deschiderea bazelor de date metoda utilizată este *CreateObject*.

- clase și obiecte de tip **Recordset** - sunt seturi de înregistrări definite de tabelele dintr-o bază de date sau rezultate din interogări făcute asupra acelei baze de date. Pentru crearea de interogări VBScript permite utilizarea de instrucțiuni SQL.

Următoarea secvență de cod ilustrează utilizarea celor două obiecte de mai sus pentru deschiderea unui set de înregistrări care conţine toate câmpurile listate într-un tabel (*table_name*) inclus într-o bază de date **wapdb.mdb.**

```
<%Option Explicit%>
<%
l_no                                    =
Request.QueryString("lecture")
Dim   adoCon,   rs,   Query,   Success,   l_no,
cuv_cheie
Set adoCon = Server.CreateObject("ADODB.Connection")
adoCon.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=" & Server.MapPath("wapdb.mdb")
Set  rs =
Server.CreateObject("ADODB.Recordset")
Query = "select content from lectures " & "where lecture_no =
'"&l_no&"'" rs.Open Query,adoCon
%>
```

În urma efectuării unei interogări , un obiect de Recordset poate conţine mai multe înregistrări; accesarea individuală a acestora se face utilizând metode predefinite (*MoveNext, MovePrevious*). Identificarea primei/ultimei înregistrări poate fi efectuată folosind de asemenea metode predefinite *BOF(beginning of file)/EOF (end of file)*. Atunci când se utilizează obiecte **Recordset**, o practică comună constă în a distruge obiectele create atunci când acestea nu mai sunt necesare. Secvenţa de comenzi VBScript care efectuează acest lucru este:

```
<% rs.Close
set rs = nothing %>
```

- clase și obiecte de tip **Response** – sunt utilizate pentru transferarea datelor către browserul care a formulat cererea. Rezultatele pot fi afişate la un client imediat sau numai după ce toate script-urile s-au încheiat; un obiect de răspuns utilizează în acest scop proprietatea *Buffer*. Când aceasta este setată la True:

```
<% Response.Buffer = "true" >
```

serverul web nu trimite răspunsul către client până când toate script-urile VBScript nu au fost executate. Dacă nu se dorește utilizarea de mecanisme de tip cache este necesară folosirea de instrucțiuni de tipul:

```
Response.Expires=0
```

- clase și obiecte **Session** – sunt utilizate pentru a salva datele în timpul unei sesiuni. O variabilă stocată pe un obiect de acest tip este persistentă pentru întreaga sesiune de utilizator, chiar dacă utilizatorul navighează înainte şi înapoi.
- clase și obiecte de tip **Request** – sunt utilizate pentru a citi local variabile și parametri transmiși de către browser către server. Secvența de instrucțiuni

```
<% Dim s
s = Request("var1")
%>
```

permite citirea într-o variabilă locală (var1) transmise de către un browser către server.

3.7.3 Exerciții

1) Analizați conținutul script-ului ASP din anexa 2. Pe baza explicațiilor anterioare şi folosind resursele disponibile la adresa https://msdn.microsoft.com/en-us/library/aa286483.aspx, explicați rolul fiecărei instrucțiuni.

2) Lansați în execuție un browser web pe computerul desktop şi, respectiv, pe emulatorul Android. Accesați următorul URL:http://193.226.17.162/interog.asp?lecture=1. Modificați valoarea parametrului transmis serverului (*lecture*) la 2, 3 și 4; vizualizați și interpretați rezultatele obținute.

3.7.4 Script-uri de partea client în HTML5

Script-urile de partea client sunt de obicei utilizate pentru a implementa logica procedurală în pagini HTML5 şi pentru a modifica sau pentru a filtra conținutul unei pagini web returnate.

Pentru definirea unui script HTML5 utilizează o etichetă dedicată **<script>**. Elementul script poate conține parametrii asociați (de exemplu variabile) sau poate indica o secvență de instrucțiuni externă folosind un atribut dedicat (**src**).

Figura următoare ilustrează o pagină HTML în care conținutul paginii web HTML5/CSS3 din figura 6 este modificat utilizând un script pe partea de client. Script-ul în sine utilizează o metodă predefinită **getElementById** care se aplică unui obiect de tip document adică paginii web în sine. Eticheta de tip paragraf **<p>** care este utilizată pentru afișarea unui text are un atribut suplimentar (**id**) care este permite referențierea acesteia și modificarea conținutului acesteia. Același efect ar putea fi realizat prin utilizarea unei funcţii externe.

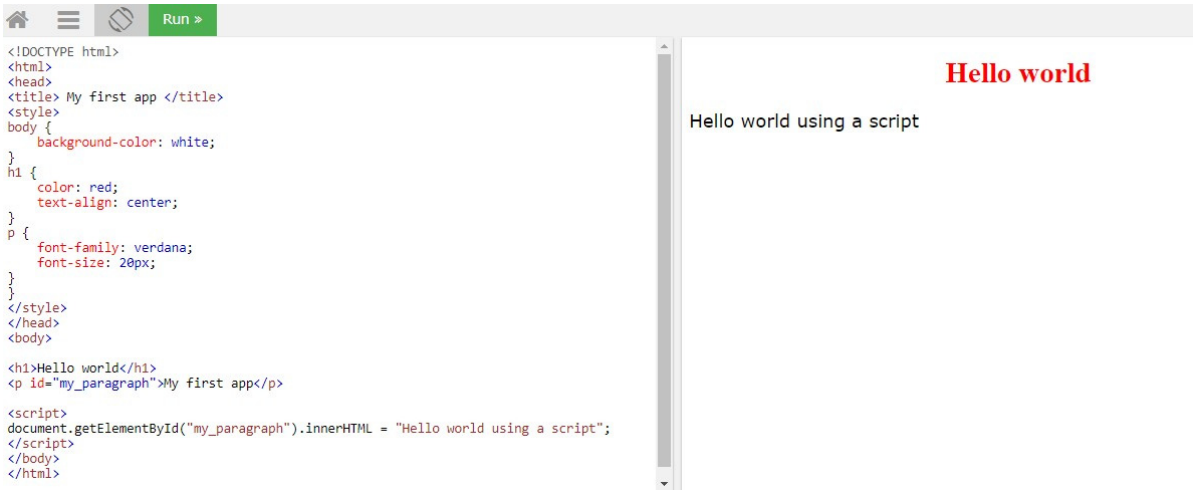


Fig.7 Exemplu de script pe partea de client

Fig.8 Exemplu de script pe partea de client utilizând funcții

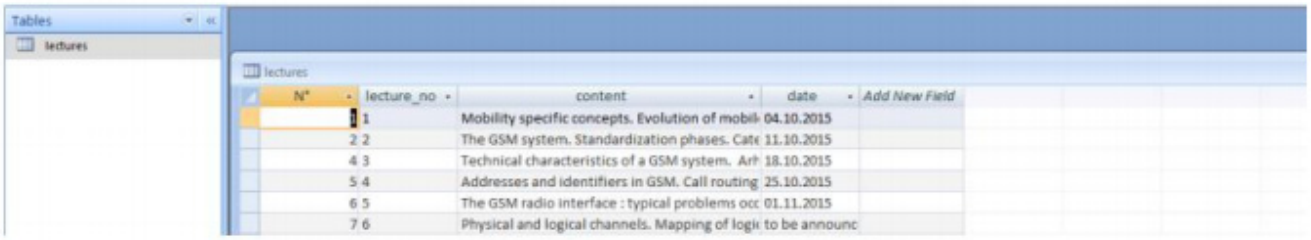
Întrebări

1) *Testați secvența ilustrată în figura de mai sus folosind emulatorul de terminal mobil Android.*

2) *Proiectați o pagină web care să utilizeze eticheta **<select>** pentru a permite selectarea unei opțiuni din 4 posibile "Lecture 1", "Lecture 2", "Lecture 3" şi "Lecture 4". Adăugați paginii create etichete de tip **<button>** și **<p>** . Adăugaţi secvența de cod care să permită modificarea dinamică a textului afișat de eticheta <**p>,** funcție de opțiunea selectată.*

3) *Utilizând rezultatele de la punctul precedent, scrieți un script care să permită afişarea informațiilor legate de aspectele tratate în cursurile de la disciplina Comunicații mobile (Indicație: utilizați obiecte de tip **XMLHttpRequest** pentru plasarea unei cereri de tip GET către script-ul ASP prezentat în § 3.7.3). Formatați pagina folosind CSS.*

Anexa 1. Conținutul bazei de date wapdb.mdb



Anexa 2 Script-ul ASP interog.asp

```
<% @LANGUAGE="VBSCRIPT"%>
<%Option Explicit%>
<%
l_no = Request.QueryString("lecture")
Dim adoCon, rs, Query, Success, l_no, cuv_cheie
Set adoCon = Server.CreateObject("ADODB.Connection") adoCon.Open
"Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" &
Server.MapPath("wapdb.mdb")
Set rs = Server.CreateObject("ADODB.Recordset")
Query = "select content from lectures " & "where lecture_no = '"&l_no&"'"
rs.Open Query,adoCon
Response.Buffer = True
Session("keywords") = rs("content")
Response.Write rs("content") rs.Close
set rs = nothing
%>
```

Lucrarea 4 Protocoale și proceduri de semnalizare în GSM

4.1 Introducere

Specificațiile tehnice GSM/GPRS/UMTS (http://www.3gpp.org) stabilesc regulile necesare pentru schimbul de mesaje de semnalizare (*proceduri de semnalizare*) ce oferă suportul pentru mobilitatea terminalului. Indiferent de tipul rețelei, procedurile suport sunt diferite și depind de de starea curentă a terminalului mobil. Sistemul GSM definește următoarele stări posibile pentru o stație mobilă:

- **inactivă (idle)–** corespunde situației în care un terminal mobil nu are un canal fizic dedicat alocat
- **activă (dedicated) –** în acest caz un terminal mobil are un canal fizic dedicat (SDCCH+SACCH) sau TCH+SACCH
- **atașată/detașată** – corespund situațiilor în care un terminal mobil este pornit sau, respectiv, oprit

În timp ce pentru starea inactivă suportul pentru mobilitatea terminalului este oferit de **proceduri de actualizare a localizării** pentru starea activă sunt folosite **mecanisme de transfer ale apelurilor** (**handover**/ handoff în sisteme non-GSM).

Procedurile de actualizare a localizării au drept scop înscrierea înscrierea în VLR a locației curente a terminalului mobil (cu o rezoluție dată de o arie de localizare etichetată printr-un identificator unic–LAI) și, dacă noua arie de localizare este gestionată de un alt echipament de tip MSC/VLR, înscrierea adresei acestuia la nivel HLR. Acest tip de procedură este inițiată întotdeaua de către o stație mobilă.

Mecanismele de transfer ale apelurilor sunt inițiate de către BSC (plăcile DTCC pentru BSC-uri Alcatel) și sunt de tip **MAHO** (Mobile Assisted Handover). În acest tip de handover deciziile luate de către rețea se bazează pe **rapoarte cu măsurători** înaintate de către stațiile mobile (downlink TCH/SDCCH și frecvențe baliză indicate de către rețea) și de către stația de bază curentă (uplink TCH/SDCCH). Rapoartele cu măsurători includ două tipuri de măsurători: **RXLEV** – nivel putere recepționată, **RXQUAL** – calitatea legăturii curente (doar TCH/SDCCH) și sunt indicate rețelei în mod indexat conform tabelelor 1 și 2.

RXLEV	Nivel putere semnal recepționat [dBm]
0	<-110
1	-109
2	-108
. . .	
63	>-48

Tabelul 1 Valori indexate RXLEV

RXQUAL	BER
0	$<2*10^{-3}$
1	$2*10^{-3}$-$4*10^{-3}$
2	$4*10^{-3}$-$8*10^{-3}$
3	$8*10^{-3}$-$1.6*10^{-2}$
4	$1.6*10^{-2}$-$3.2*10^{-2}$
5	$3.2*10^{-2}$-$6.4*10^{-2}$
6	$6.4*10^{-2}$-$1.28*10^{-1}$
7	$>1.28*10^{-1}$

Tabelul 2 Valori indexate RXQUAL

Deciziile luate de un BSC sunt specificate în [1] și au la bază următoarele criterii (în ordinea priorității):

- intracell (i.e. Intra-timeslot): valoarea RXLEV relativ mare și RXQUAL de valoare mare; corespunde unei situații de interferență puternică ce se manifestă pe canalul temporal (TS) alocat și presupune comutarea canalului dedicat într-un alt TS
- RXQUAL uplink și downlink de valori mari; canalul curent este transferat într-o altă celulă
- RXLEV uplink și downlink de valori mici; canalul curent este transferat într-o altă celulă

4.2 Protocoale de semnalizare pe interfața radio

Protocoalele de semnalizare pe interfața radio sunt organizate conform următoarei stive:

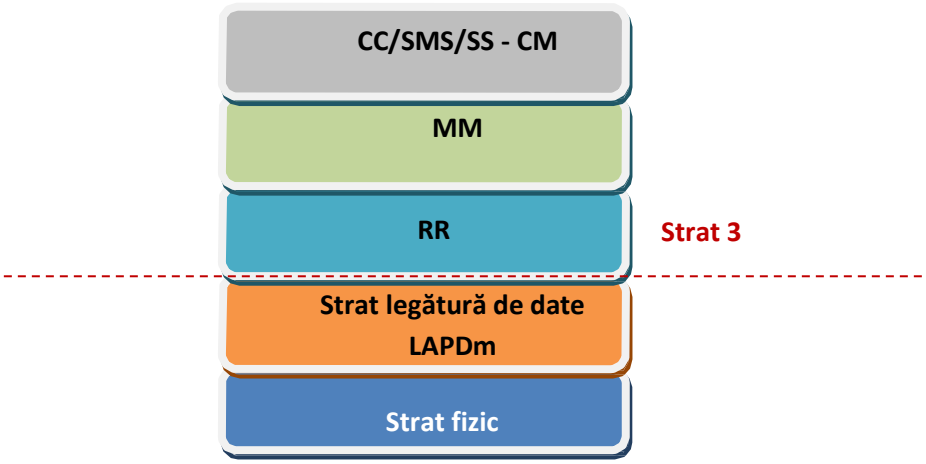


Fig.1 Stiva protocoalelor de semnalizare pe interfața radio GSM

4.2.1 RR- Radio Resource Management

Familie de mesaje dedicate pentru gestiunea resurselor radio ce definesc:

– tipul şi formatul mesajelor schimbate între statia mobila şi retea pentru a stabili un canal dedicat punct-la-punct ce permite dialogul dintre staţia mobilă şi reţea (**conexiune RR**)

– tipul şi formatul formatul mesajelor transmise pe BCCH, SCH, FCCH

– tipul şi formatul mesajelor efectuate pe AGCH/PCH /RACH

- tipul și formatul mesajelor transferate între un mobil în stare activă și rețea pentru a permite implementarea de mecanisme de transfer ale apelurilor pentru mobilitatea terminalului

Echipamentele GSM ce implementeaza familia de protocoale RR sunt MS, BTS, BSC ; toate procedurile RR (schimbul de mesaje RR) au loc doar în interiorul rețelei de acces radio.

4.2.2 MM-Mobility Management

MM este protocolul GSM responsabil de gestiunea mobilităţii ; implementează proceduri de gestiune a locației utilizatorilor mobili (pentru terminale în stare inactivă) precum şi proceduri legate de securitate şi confidenţialitate (autentificare, stabilirea modului de lucru criptat, etc).

Orice procedură MM necesită efectuarea în prealabil a unei proceduri de tip RR pentru alocarea unui canal dedicat (SDCCH+SACCH asociat) pe care transferul mesajelor definite de acest protocol are loc în mod transparent (rețeaua de acces nu interpretează mesajele ci oferă doar suportul necesar transferului acestora între MS și MSC/VLR).

Principalele tipuri de proceduri MM sunt:

- actualizarea localizarii
- atasare/detasare de la retea (IMSI attach/detach)
- autentificare, criptare etc

Anumite proceduri de tip CM necesită efectuarea în prealabil a unor proceduri de tip MM (în procedurile de stabilire a unui apel utilizatorul este autentificat înainte de efectuarea apelului).

4.2.3 CM-Connection Management

Familie de protocoale ce definește formatul și tipul mesajelor folosite în dialogul MS-rețea nucleu NSS pentru:

- stabilire/terminare a unui apel –**CC (Call Control)**
- transferul mesajelor scurte –**SMS(Short Message Service)**
- activare servicii suplimentare –**SS (Supplementary Services)**

4.3 Formatul unui mesaj de strat 3 pe interfața radio

Indiferent de familia de protocoale căreia îi aparține un mesaj de strat 3 are formatul indicat în tabelul 3 [2].

Octet 1	Transaction identifier/Skip indicator	Protocol discriminator
Octet 2	Message	
Octet 3,...	Information	

Tabelul 3 Formatul unui mesaj de strat 3 pe interfața radio

Câmpul **Protocol Discriminator** este codat pe 4 biți și identifică protocolul care a generat mesajul conform unor valori prestabilite indicate în [2]. **Tipul mesajului** este codat pe 8 biți și identifică tipul mesajului în cadrul aceleiași familii (un anumit mesaj MM,CM sau RR; ex: MEASUREMENT_REPORT trimis de BTS 00101010). Câmpul **Transaction Identifier** permite efectuarea unor semnalizări CM concurente și este setat pe 0000 pentru mesaje de tip MM și RR.

4.4 Principalele tipuri de mesaje de semnalizare

4.4.1 Mesaje RR

Mesaje difuzate

SYSTEM INFORMATION TYPE 1 - indică informații de uz general : reguli pentru accesul aleator, lista purtătoarelor folosite în celulă, purtătoare BCCH

SYSTEM INFORMATION TYPE 2 - indică lista purtătoarele corespunzând frecvențelor baliză din
celulele vecine ce trebuie monitorizate

SYSTEM INFORMATION TYPE 3 - parametrii necesari identificării ariei de localizare: LAC(Location Area Code) + MCC (Mobile Country Code)+ MNC(Mobile Network Code). Cei trei identificatori concatenați formează LAI (Location Area Identifier) – un identificator unic în aria de servicii GSM pentru orice arie de localizare. Același tip de mesaj indică modul de organizare a canalelor de control pe TS0; o valoare implicită de 0 presupune o organizare de tipul pe TS0 în cadre TDMA consecutive:

FSBBBBPPPPFSPPPPPPPPPFSPPPPPPPPPFSPPPPPPPPPFSPPPPPPPPPI F-

FCCH, S- SCH, B-BCCH, P –RACH (uplink), AGCH/PCH (downlink), I-Idle

SYSTEM INFORMATION TYPE 4 – repetare a informațiilor incluse în mesajele anterioare + informații legate de mapare a mesajelor difuzate în celulă

Toate mesajele de tip SYSTEM INFORMATION TYPE 1-4 sunt transmise pentru stații mobile aflate în stare inactivă.

SYSTEM INFORMATION TYPE 5 – doar pentru mobile în stare activă, include lista purtătoarelor învecinate ce trebuie monitorizate pentru un eventual handover (campul *Channels)*. Acest tip de mesaj este transmis pe canalul de semnalizare lent asociat (SACCH) unui canal de tip TCH sau a unui canal de tip SDCCH.

SYSTEM INFORMATION TYPE 6 – parametri CI în stare activă- Cell Identity, LAI, indicații folosire transmisie discontinuă etc.

Standardul prevede și alte tipuri de mesaje de tip SYSTEM INFORMATION, descrierea completă a

acestora poate fi consultată în [2].

Mesaje de paging și alocare de canale dedicate

PAGING REQUEST TYPE 1,2,3 – indică un apel destinat stației mobile. Stația mobilă adresată este identificată prin TMSI sau prin IMSI (dacă TMSI nu este disponibil). Mesajele de paging se transmit ciclic conform modulului de mapare al canalelor ilustrat mai sus.

CHANNEL REQUEST – cerere stabilire canal ca răspuns la paging sau la inițiativa stației mobile. Cererea de canal este codificată pe 8 biți din care 3 sunt folosiți pentru indicarea motivului accesului și restul de 5 pentru un număr aleator ales de către stația mobilă. Mesajul se transmite pe AGCH.

IMMEDIATE ASSIGNEMENT - folosit de către rețea în direcția downlink pentru asignarea unui canal. Mesajul indică stației mobile:

- **un canal de control dedicat SDCCH**. Parametrii acestuia număr canal, interval temporal TS, etichetă secvență de antrenare, număr cadru TDMA sunt incluși în mesaj
- **un canal dedicat SDCCH precum și parametri pentru saltul de frecvență pe SDCCH.** Setul de parametri cuprinde:
  MA –Mobile Allocation – lista frecvențelor din celulă pe care se efectuează saltul de
  frecvențe. Frecvențele efective se cunosc de pe canale de difuzare al celulei.
  MAIO – Mobile Allocation Index Offset – purtătoare inițială pentru saltul de frecvență
  HSN – Hopping Sequence Number – o valoare din 64 posibile ce definește modul în care se efectuează saltul de frecvență. O valoare de 0 presupune utilizarea ciclică a frecvențelor ce sunt utilizate în celulă (cunoscute din mesaje de tip SYSTEM INFORMATION TYPE 1)
- **valoarea TA (Timing Advance) –** comenzi de avansare temporală pentru evitarea interferențelor datorate distanței variabile dintre stația mobilă și stația de bază
- **numărul aleator ales de către stația mobilă** – pentru evitarea eventualelor coliziuni

PAGING RESPONSE - folosit de către mobil pentru a indica că a recepționat mesajul de paging. Stația mobilă folosește acest tip de mesaj pentru a indica rețelei diverse caracteristici tehnice pe care le implementează (tip algoritm criptare, benzi suportate GSM/DCS, clasă putere, SMS, capabilități 3G etc). Mesajul include de asemenea și un număr (*Ciphering Key Sequence Number CKSN)*, ce indică cheia de criptare stocată pe cartela SIM.

Classmark change/Classmark enquiry – mesaje dedicate pentru negocierea/modificarea caracteristicilor stației mobile.

MEASUREMENT REPORT - mesaje folosite pentru handover. Sunt transmise pe SACCH și au la

bază un ciclu de măsurare de 0.5 s. Pentru toate purtătoarele indicate de mesajele SYSTEM INFORMATION TYPE 5 mobilul include în măsurători parametrii menționați mai sus: RXQUAL și RXLEV. Pentru celulele învecinate valorile măsurate sunt indexate prin valorile BSIC și numărul de ordine al purtătoarelor indicate în lista transmisă în mesaje SYSTEM INFORMATION. Măsurătorile sunt efectuate în 2 moduri (full – toate intervalele temporale, sub – doar intervalele temporale în care stația mobilă este activă dacă se folosește transmisia discontinuă).

ASSIGNMENT COMMAND – folosit în direcția downlink pentru alocarea unui canal de trafic. Mesajul include descrierea completă a canalului alocat (TCH): time slot, ARFCn, parametrii salt de frecvență, nivelul de putere inițial ce va fi folosit de către stația mobilă. Mesajul este confirmat de către stația mobilă pe canalul alocat în mod FACH printr-un mesaj RR de tip **ASSIGNEMENT COMPLETE** Canalul alocat poate fi folosit pentru mesaje de tip CM/MM.

HANDOVER COMMAND – comandă pentru transferul unui canal dedicat într-o altă celulă sau pe un alt interval temporal în aceeași celulă. Mesajul include:

- purtătoarea ce transportă canalul BCCH în noua celulă (ARFCn)
- descriere canal nou alocat (TS, ARCFn, tip canal: TCH+SACCH, cod secvență de antrenare pentru noua celulă,BSIC), folosirea sau nu a unor salve de acces. Noul canal este activat în prealabil în mod recepție de către BSC pe un TRE, într-un BTS responsabil de acoperirea radio în noua celulă.
- referință handover – număr aleator alocat de BSC

HANDOVER ACCESS – include referința pentru handover, trimis în direcția uplink pentru a permite unui BTS să estimeze valoarea TA. Recepția corectă este confirmată în stratul legătură de date de către un BTS

PHYSICAL INFORMATION – (opțional) – transfer valoare TA către stația mobilă

HANDOVER COMPLETE – confirmarea de către stația mobilă a faptului că procedura a avut loc cu succes.

4.4.2 Mesaje MM

Transferul mesajelor MM necesită existența unui canal de semnalizări dedicat (SDCCH).

LOCATION UPDATING REQUEST – transmis pe canalul SDCCH în direcția uplink; include identificatorul LAI stocat pe cartela SIM și identificatorul TMSI vechi. Noua valoarea LAI va fi adăugată mesajului de către BSC. Modul de mapare celule <-> identificatori LAI este stabilit prin operații de tip O&M. Pornind de la identificatorii trimiși de către stația mobilă, la nivel MSC/VLR, se determină identitatea GSM a abonatului (IMSI) și parametri de autentificare via HLR sau din vechiul VLR.

LOCATION UPDATING ACCEPT – include noul cod LAI ce va fi înscris pe cartela SIM și folosit în proceduri ulterioare de actualizare a localizării. Opțional, dacă MSC/VLR se schimbă, mesajul

include un nou identificator TMSI

AUTENTHICATION REQUEST - mesaj ce include un numar aleator pe 128 de biți transmis de către rețea (RAND). La primirea mesajului stația mobilă calculează o semnătură (SRES) conform unui algoritm predefinit (A3). În același mesaj rețeaua include un nou număr CKSN pentru indexarea pe SIM a noii chei de criptare, calculată cu un algoritm predefinit A8.

AUTENTHICATION RESPONSE - răspuns al stației mobile ce include semnătura (SRES) calculată la primirea mesajului anterior.

CIPHERING MODE COMMAND – mesaj transmis în direcția downlink ce inițiază transferul criptat de informație pe interfața radio. Mesajul include indicații cu privire la textul în clar ce va fi criptat de către stația mobilă (tipic IMEI); anterior procedura de decriptare este inițializată la nivel BTS cu aceeași parametri.

CIPHERING MODE COMPLETE – mesaj transmis de către stația mobilă ce include textul în clar indica în mesajul precedent. Mesajul este criptat de către MS și decriptat de către BTS.

4.4.3 Mesaje CM/CC

SETUP- mesaj transmis în direcția downlink de către rețea pentru a determina stația mobilă să demareze procedura de stabilire a unui apel de tip MT sau în direcția uplink de către stația mobilă pentru a iniția procedura de stabilire a unui apel de tip MO. Mesajul include descrierea serviciului (codec voce, rată de transfer serviciu de date) solicitat precum și numărul părții apelante/apelate funcție de direcția donwlink/uplink în care se transmite mesajul.
CALL CONFIRMED – mesaj transmis de către stația mobilă apelată prin care se confirmă faptul că apelul poate fi acceptat cu indicarea caracteristicilor tehnice ale stației mobile pentru serviciul solicitat (tipuri codoare de voce, parametrii QoS etc). Rețeaua poate selecta unul dintre codoarele de voce indicate de către stația mobilă.
ALERTING – mesaj prin care se indică părții corespondente că partea apelată a fost alertată și că apelul inițiat prin SETUP poate fi deservit.
CONNECT/CONNECT ACKNOWLEDGE – indicații cu privire la faptul că ambii utilizatori au acceptat
apelul.
CM SERVICE REQUEST/CM SERVICE ACCEPT – folosite în procedura de stabilire a unui apel de tip MO pentru a indica/confirma serviciul cerut. În cerere stația mobilă indică tipul apelului (apel de voce de tip MO, apel de urgență, SMS, apel de date), caracteristicile tehnice pentru procesare pe interfața radio, identitatea (TMSI) etc.
CALL PROCEEDING - folosit de către rețea pentru a indica stației mobile că nici o altă cerere nu poate fi procesată până la deservirea cererii curente.

4.5 Proceduri de semnalizare în stare inactivă (idle)

Mesajele de semnalizare transmise pe interfața radio pot fi analizate folosind aplicații

dedicate precum TEMS Investigation sau Qualcomm QXDM/QCAT.

Interfața grafică a aplicației este cea ilustrată în Fig.2 pentru prima aplicație; mesajele de semnalizare din secțiunile următoare au fost achiziționate folosind un telefon mobil dedicat, compatibil TEMS. Pentru lucrarea de laborator mesajele de semnalizare au fost stocate în fișiere cu extensia ***log*** (Fig.3); fiecare fișier corespunde unui scenariu ce include diverse proceduri de semnalizare.

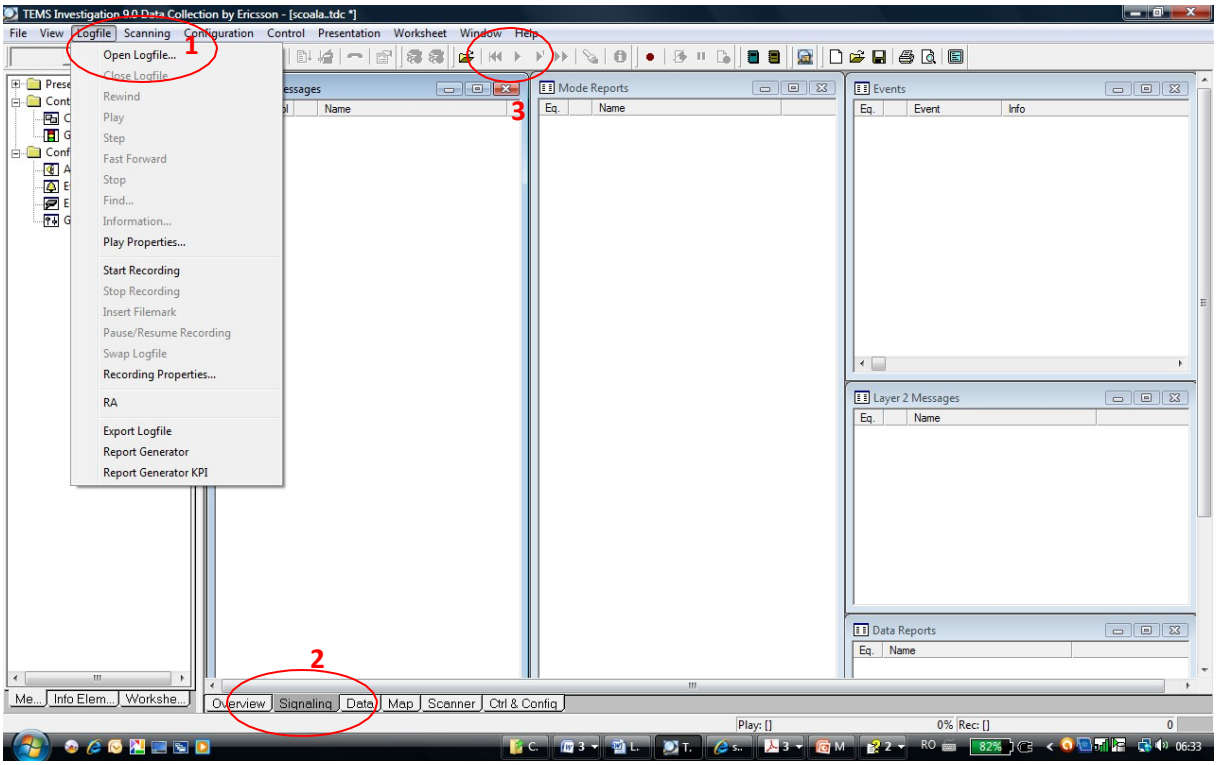


Fig.2 Interfața cu utilizatorul a aplicației TEMS investigation

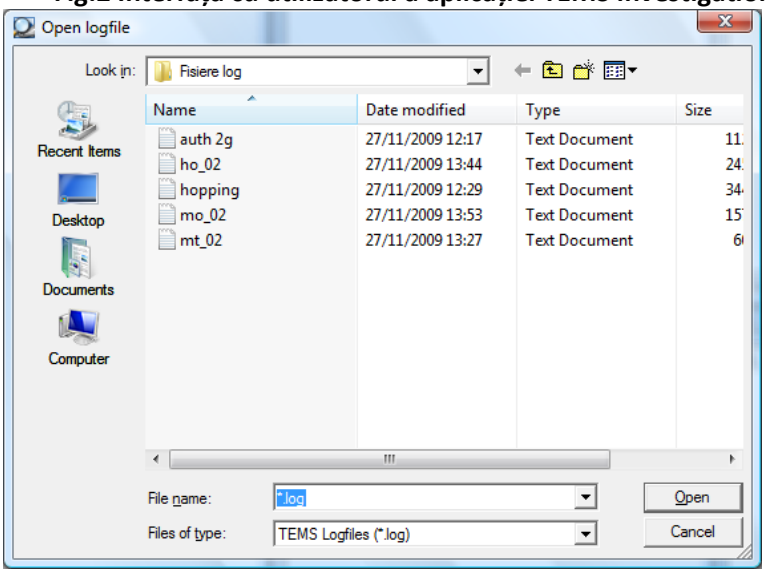


Fig.3 Fișiere .log cu mesaje de semnalizare.

Prin selectarea unui anumit fișier acesta poate fi redat dinamic (Fig.2 (1)) cu ilustrarea principalelor mesaje de semnalizare și a stărilor corespunzătoare ale stației mobile.

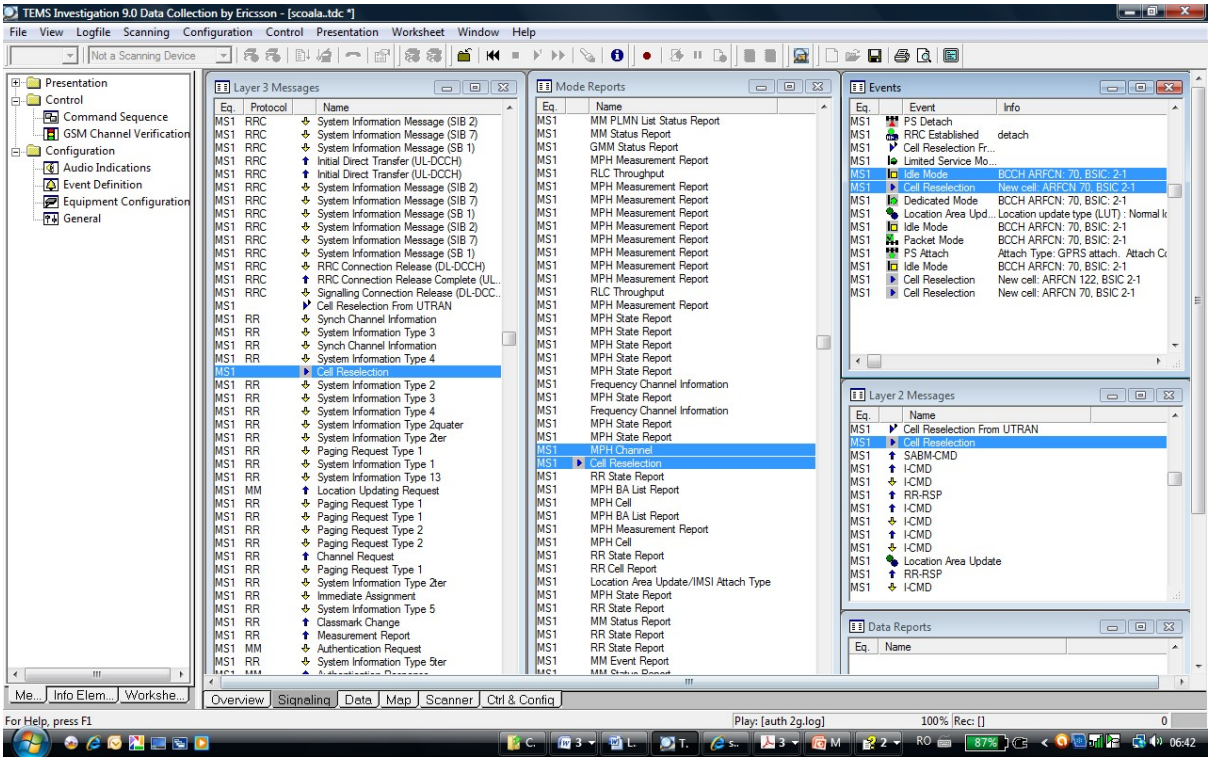
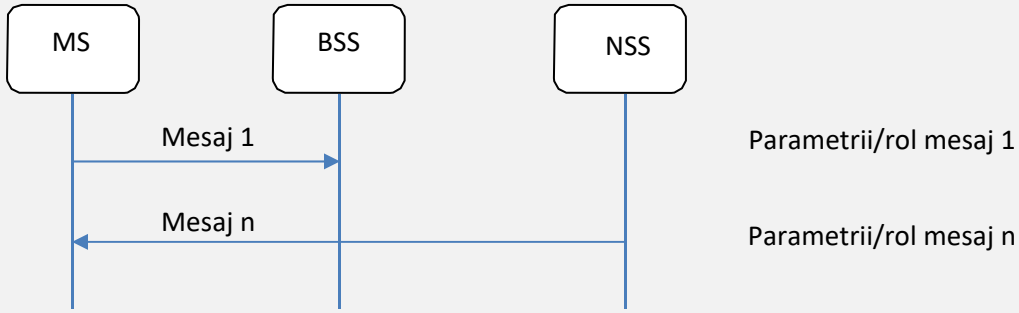


Fig.4 Mesaje de semnalizare (opțiunea Signaling Fig.2 (3))

Întrebări

1. *Folosind aplicația TEMS Investigation deschideți fișierul **auth_2g.log.** Acesta corespunde unui scenariu de actualizare a localizării în rețeaua Orange România.*
2. *Analizați tipurile de mesaje schimbate între rețea și stația mobilă și reprezentați pe o diagramă simplificată (ilustrată mai jos) succesiunea temporală a principalelor mesaje de semnalizare cu evidențierea parametrilor transferați și explicarea funcțiilor îndeplinite de fiecare mesaj*



3. Folosiți explicațiile din paragraful 4.4 și [2] și considerați ca stare inițială a stației mobile cea ilustrată în Fig.3 (Idle mode).

4. Ce modificări ar apărea în succesiunea mesajelor de semnalizare dacă procedura s-ar efectua într-o rețea 3G via UTRAN?

4.6 Proceduri de semnalizare în stare activă (dedicated)

Fișierul **ho_02.log** include mesajele de semnalizare asociate unei proceduri de handover pentru macheta de laborator la transferul unui apel dintr-o celulă în celula învecinată.

Întrebări

1. Similar procedurii indicate în 4.5 analizați tipurile de mesaje schimbate între rețea și stația mobilă și reprezentați pe o diagramă simplificată (ilustrată mai jos) succesiunea temporală a principalelor mesaje de semnalizare cu evidențierea parametrilor transferați și explicarea funcțiilor îndeplinite de fiecare mesaj.

2. Care este criteriul folosit pentru transferul apelului?

4.7 Apeluri de tip MT

Fișierul **mt_02.log** include mesajele de semnalizare asociate unei proceduri pentru stabilirea unui apel de tip MT folosind macheta de laborator.

Întrebări

1. Folosind ca stare de pornire starea "Cell Reselection", similar procedurii indicate în 4.5, analizați tipurile de mesaje schimbate între rețea și stația mobilă și reprezentați pe o diagramă simplificată (succesiunea temporală a principalelor mesaje de semnalizare cu evidențierea parametrilor transferați și explicarea funcțiilor îndeplinite de fiecare mesaj.
2. Care dintre mesajele indicate în secțiunea 4.4 intervin în procedura de semnalizare? Care este protocolul care le definește?
3. Care este numărul de handover-uri care apar explicit în procedura de semnalizare? Care sunt criteriile asociate de efectuare?
4. Ce modificări ar apărea dacă apelul ar fi deservit de o rețea 3G?
5. Apelul este efectuat între 2 stații mobile aflate în aceeași celulă sau în celule diferite?

4.8 Apeluri de tip MO

Fișierul **mo_02.log** include mesajele de semnalizare asociate unei proceduri pentru stabilirea unui apel de tip MO folosind macheta de laborator descrisă în lucrarea 1.

Întrebări

1. Folosind ca stare de pornire starea indicată în, similar procedurii indicate în 4.5, analizați tipurile de mesaje schimbate între rețea și stația mobilă și reprezentați pe o diagramă succesiunea temporală a

principalelor mesaje de semnalizare cu evidențierea parametrilor transferați și explicarea funcțiilor îndeplinite de fiecare mesaj.

2. Apelul este efectuat între 2 stații mobile aflate în aceeași celulă sau în celule diferite?

Fișierul **hopping.log** include mesajele de semnalizare asociate unei proceduri pentru stabilirea unui apel de tip MO folosind rețeaua Orange România.

Întrebări

1. Analizați comparativ diferențele dintre apelul realizat în macheta de laborator și cel dintr-o rețea reală. 2. Explicați rolul fiecărei proceduri de semnalizare suplimentare.

Bibliografie

[1]. ETSI/GSM Recommendations 05.08, "Digital cellular telecommunications system (Phase 2+); Radio subsystem link control"
[2].ETSI/GSM Recommendations 04.08 "Digital cellular telecommunications system (Phase 2+); Mobile radio interface layer 3 specification", version 8.4.0, 1999.

Lucrarea 5 Aplicații Android

5.1 Introducere

Android este o platformă software care permite dezvoltarea aplicațiilor mobile pe smartphone-uri echipate cu versiune modificată a sistemului de operare Linux.

Tehnologia a fost propusă inițial de Google şi ulterior dezvoltată de Open Handset Alliance, o corporaţie formată din actori cheie pe piaţa IT&C, care are drept obiectiv principal promovarea inovării în domeniul aplicațiilor mobile prin folosirea de tehnologii open-source [1].

Pentru dezvoltarea aplicaţiilor Android, un mediu dedicat de programare (*SDK- Software Developpement Kit*) Android (*Android SDK*) [2] este utilizat uzual în conjuncție cu un mediu integrat de dezvoltare integrat (*IDE- Integrated Developpement Environment*). Exemple de IDE-uri : Eclipse, NetBeans, Android Studio.

Limbajul utilizat uzual pentru programarea pe platforma Android este Java (există posibilitateavde a utiliza însă și alte limbaje cum ar fi *Kotlin* sau *C++)* iar mașina virtuală Java poartă numele de *ART (Android Runtime)* (utilizată de versiuni Android mai noi de 5.0) sau *Dalvik (*versiuni mai vechi). Aplicațiile dezvoltate pentru mașina virtuală Dalvik sunt compatibile cu cele dezvoltate pentru ART.

Versiunea curentă de Android este Android 9 Pie.

5.2 Structura unei aplicații Android

Aplicaţiile Google Android sunt dezvoltate utilizând programarea orientată pe obiecte (*OOP*). Orice aplicație include una sau mai multe componente de următoarele tipuri:

Activity – clasă asociată cu o fereastră/interfață grafică care poate fi populată cu diferite tipuri de alte componente de tip *UI (User Interface)* cum ar fi meniuri, liste, casete text, spinner-e etc. Utilizarea de ferestre multiple necesită mai multe instanțe ale acestei clase, o aplicație Android fiind uzual formată din una sau mai multe componente de tip Activity.

Service – componentă care include secvenţe de cod care rulează în background (posibil în alte thread-uri) și nu oferă o interfață grafică.

Content provider – componentă Android care permite transferul și partajarea de date între diverse aplicații. Datele pot fi stocate în baze de date (tipic SQlite), citite de pe Internet sau stocate în format sistem de fișiere.

Broadcast receiver – componentă ce permite interceptarea evenimentelor sistem și a mecanismelor de tip ***Intent*** ce permit transferul asincron de mesaje între elemente de tip Activity și/sau Service, sau instanțierea a altor obiecte de tip Activity.

Toate clasele utilizate de o aplicație şi proprietăţile lor asociate trebuie să fie declarate în format XML într-un fişier numit ***AndroidManifest.xml*** inclus în proiect.

5.3 Dezvoltarea de aplicații Android folosind mediul de dezvoltare Eclipse

Conform celor menționate mai sus , medii de dezvoltare de tip IDE sunt de obicei utilizate în conjuncție cu un SDK Android. Secțiunea prezintă doar modalitatea de utilizare a mediului *Eclipse* pentru dezvoltarea de aplicații Android. Fig.1 explicitează etapele necesare pentru configurarea unei astfel de aplicații folosind versiunea de Android 4.2 (*JellyBean*) care este instalată pe toate calculatoarele din laborator.

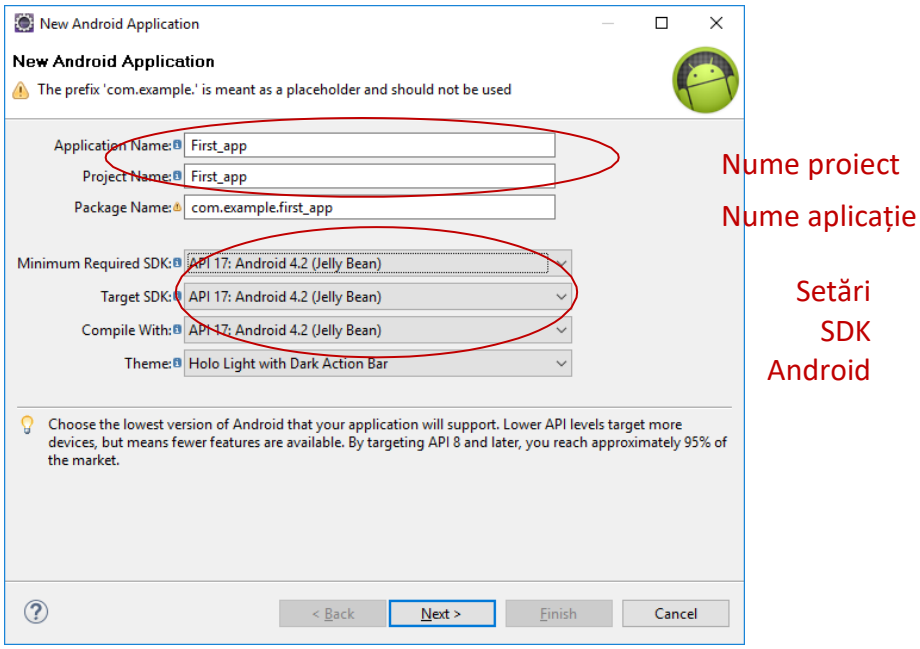
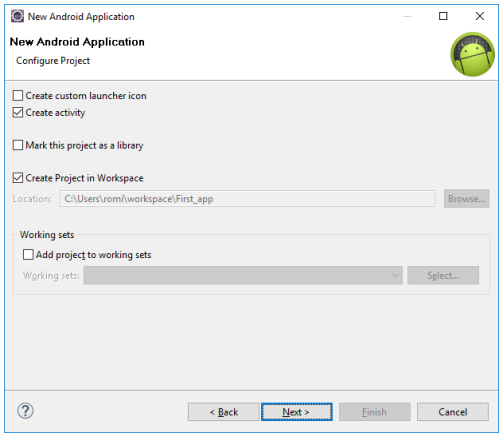
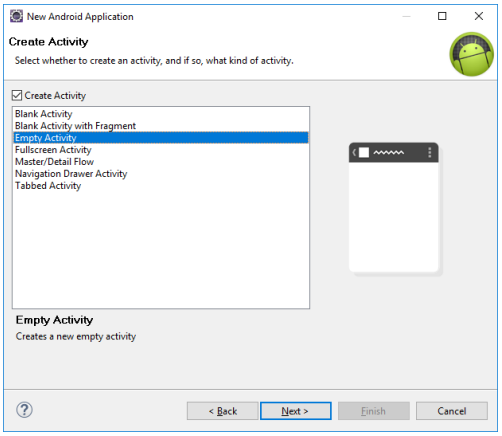


Fig.1 Setări aferente unei aplicații Android în Eclipse

Etapele suplimentare sunt indicate în Fig.2 iar structura aplicației create automat este cea ilustrată în Fig.3.





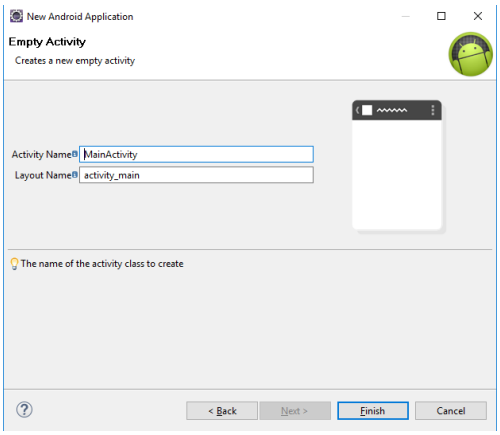


Fig.2 Etapele necesare creării unui proiect de tip *Empty Activity* folosind Eclipse

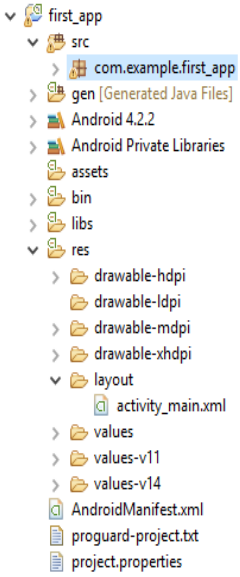


Fig.3 Structura unei aplicații Android în Eclipse

Orice proiect Android trebuie să includă un fișier cu numele ***AndroidManifest.xml*** care trebuie plasat în directorul rădăcină aferent proiectului. Pentru aplicația anterioară structura acestuia este:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.example.first_app"
      android:versionCode="1"
      android:versionName="1.0" >

      <uses-sdk android:minSdkVersion="17"
            android:targetSdkVersion="17" />

      <application android:allowBackup="true"
            android:icon="@drawable/ic_launcher"
            android:label="@string/app_name"
            android:theme="@style/AppTheme">
            <activity android:name=".MainActivity"
```

```
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Fig.4 Fișierul AndroidManifest.xml pentru aplicația creată

Proprietăţile aplicaţiei sunt specificate în format *eXtensible Markup Language (XML)*; aceste proprietăţi includ:

-numele pachetului (*package name)* ca identificator unic al aplicației (*com.example.first_app* în Fig.4);

- tipurile de componente și numele acestora utilizate în aplicație (o aplicație cu interfață grafică de tip *Activity* în Fig.4);

- permisiunile aplicației (de exemplu acces la informația de localizare, acces la Internet etc; nu este cazul în figura 4).

Fișierul **MainActivity.java** include codul Java propriu-zis; pentru aplicația creată acesta are conținutul:

```
package com.example.first_app;

import android.app.Activity;
import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Fig.5 Fișierul MainActivity.java

Cuvântul cheie ***extends***, similar aplicațiilor JME, JSE sau JME, este utilizat pentru a implementa mecanisme de moștenire Java (preluarea structurii și a comportamentului unei alte clase cu adăugarea de elemente noi, specifice).

Cuvântul cheie ***@Override*** permite supradefinirea metodelor clasei părinte.

Pentru exemplul din figura 5, metoda ***OnCreate*** este prima metodă apelată la lansarea aplicației (echivalent JME **StartApp()**). Obiectele de tip ***Bundle*** sunt folosite în Android pentru gestiunea situațiilor în care aplicațiile sunt eliberate din memorie.

Fișierul **activity_main.xml**, plasat în directorul "res/layout" permite definirea interfeței cu

utilizatorul în format XML sau, pentru anumite IDE-uri, și în format grafic. Pentru aplicația creată anterior acesta are structura următoare:

```
<RelativeLayout
                    xmlns:android="http://schemas.android.com/apk/res/android
  " xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
                    android:layout_height="match_parent"
  tools:context="${relativePackage}.${activityClass}" >

<TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="@string/hello_world" />
 </RelativeLayout>
```

Fig.6 Fișierul activity_main.xml

Fişierul XML poate fi modificat fără a fi necesară o recompilare a codului sursă; acest mod de concepere al unei aplicații este asimilabil modelului arhitectural *MVC (Model View Controller)* de dezvoltare a aplicațiilor software. Acesta divide o aplicație în trei componente *Model, View, Controller*; componenta *Model* gestionează datele aplicației, componenta *View* permite afișarea modelului și interacțiunea cu utilizatorul, iar componenta *Controller* interceptează acțiunile utilizatorului inițiate prin intermediul (tipic) touchscreen-ului. Elementele UI care pot fi utilizate într-o aplicație Android de tip *Activity* sunt definite complet în [3], printre acestea fiind:

TextView – afișarea de informații de tip text;

EditText – editarea informației de tip text;

Button – elemente UI de tip command-button pentru interacțiunea cu utilizatorul.

După compilarea aplicației elementele definite în fișierul xml sunt reprezentate prin componente individuale Android de tip *View* ce ocupă o anumită zonă de pe ecran. Apelarea metodei *onCreate* cu un parametru de tip *R.layout.layout_file_name* :

setContentView (R.layout.file_name);

permite afișarea acestora.

Toate obiectele Android de tip *View* pot fi referențiate în mod unic dacă li se asociază identificatori sub formă de numere întregi. Următoarea secvență xml permite asocierea de identificatori unici elementelor de tip UI:

android: id="@+id/elem_UI

Subșirul **@+id** indică compilatorului să creeze o nouă resursă de tip *View*; pentru exemplul de mai sus numele acestuia este *elem_UI*. Toate aceste componente de tip xml declarate sunt incluse într-

un fișier cu numele ***R.java*** generat automat în etapa de compilare. Acest fișier permite mai apoi asocierea resurselor grafice codului Java. Elementele UI astfel declarate pot fi referite în cod prin folosirea de instrucțiuni de tipul:

Class obiect_UI = (Class) findViewById(R.id.elem_UI);

Elementele de tip UI asociate tastelor programabile (meniuri) sau obiecte Android de tip *Spinner* pot fi declarate de asemenea în fișiere xml; în cazul definirii unui meniu de tip pop-up fișierul xml trebuie să fie distinct.

Aplicațiile pot fi testate folosind emulatoare definite în prealabil, cu diferite caracteristici (rezoluție, memorie etc). În cazul aplicației create anterior, rezultate posibile sunt afișate în Fig. 7.

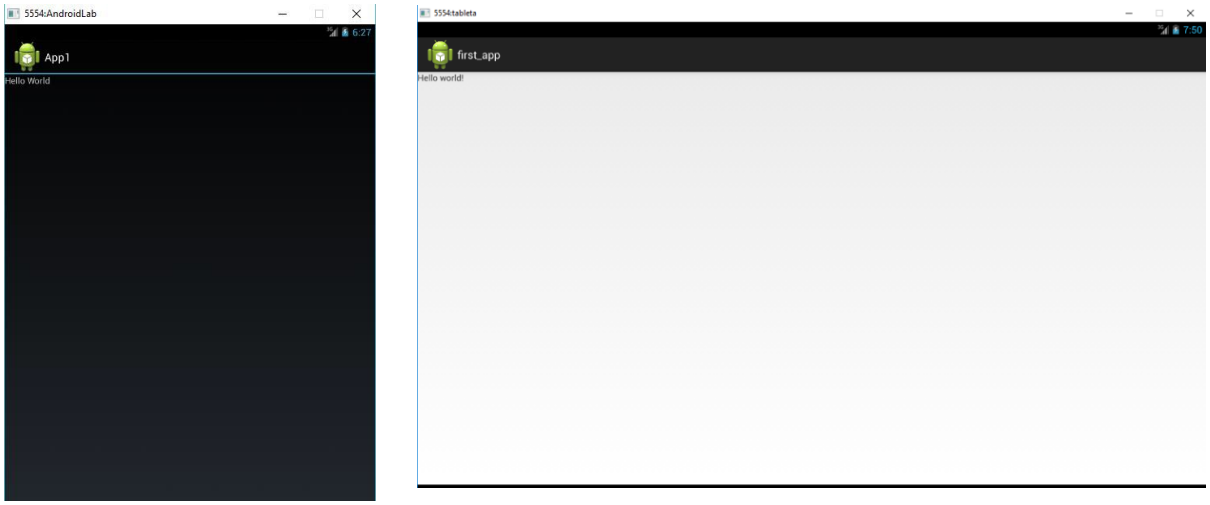


Fig.7 Emulatoare Android (AVD) – rezultate ale rulării aplicației de tip Hello world

5.4 Exerciții

1) Repetați pașii indicați mai sus pentru a creea o aplicație Android folosind mediul de dezvoltare
Eclipse.
2) Folosind [3] inserați comentarii în codul Java și xml, ilustrând rolul fiecărei instrucțiuni.
3) Modificați aplicația astfel încât să afișeze un alt text.
4) *Testați aplicația folosind mai multe emulatoare Android (Android Virtual Device).*
5) *Creați un fișier de tip .doc; inserați capturi de ecran care să ilustreze funcționarea aplicației pentru textul definit la pct. 4)*

5.5 Aplicație client-server folosind Android (client) și tehnologia Microsoft IIS (server)

5.5.1 Architectura aplicației

Arhitectura aplicației este cea indicată în Fig.8.

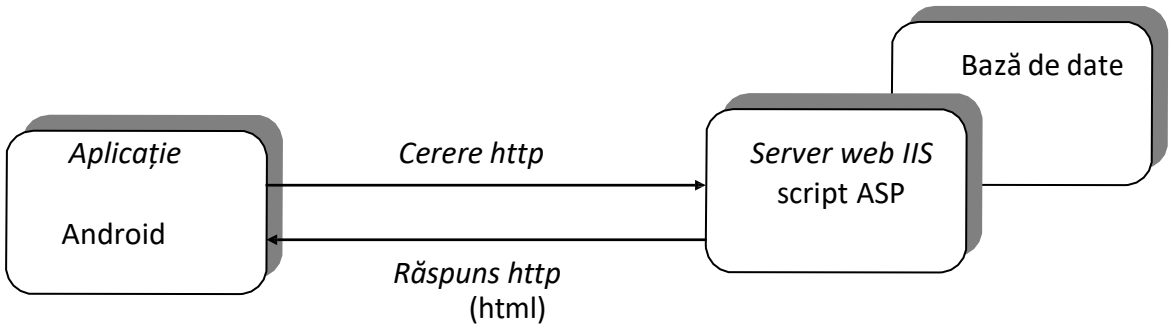


Fig.8 Arhitectura aplicației

5.5.2 Structura și conținutul bazei de date

Informațiile stocate sunt indicate în fig.9.

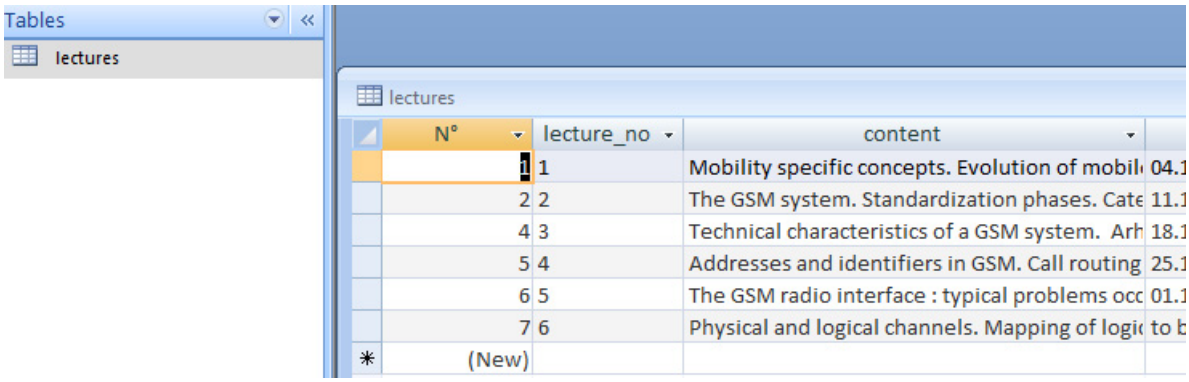


Fig.9 Structura bazei de date/informațiile stocate

5.5.3 Aplicația client Android

Elemente UI

1.Creați o aplicație Android procedând într-un mod similar cu cel indicat în secțiunea 5.3

2.Modificați fișierul *activity_main.xml* după cum urmează:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   android:id="@+id/widget29"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent "
   android:orientation="vertical"
   android:textColor="#ffffff"
```

```
  xmlns:android="http://schemas.android.com/apk/res/android"  >

<TextView    android:id="@+id/text"
        android:textStyle="bold"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffff00"
        android:textSize="32sp" android:text="Select
        URL" >
 </TextView>

 <EditText android:id="@+id/setari"
        android:layout_width="fill_parent"
        android:layout_height="60dip"
        android:capitalize="sentences" >
 </EditText> <Button android:id="@+id/ok"
   android:layout_width="fill_parent"
android:layout_height="60dip" android:text="OK" />

</LinearLayout>
```

Elementele UI folosite pentru interacțiunea cu utilizatorul sunt:

- o componentă de tip *TextView* pentru afișarea unui text prestabilit (ID Android t**ext**)
- o componentă de tip *EditText* care permite afișarea unei căsuțe de dialog pentru setarea adresei IP a serverului (ID Android **settings)**
- un buton de comandă (ID Android **OK)**

Modul de afișare a elementelor UI corespunde unui dispuneri liniare, având ca origine colțul din stânga sus al telefonului ecranului. Pentru toate elementele UI, lățimea a fost setată la dimensiunea ecranului (atributul FILL_PARENT pentru lățime) iar înălțimea lor este fie variabilă (cuvânt cheie *WRAP_CONTENT*) fie setată la o valoare implicită.

3) Integrați secvența *this.setTitle("Course content ...");* în corpul metodei *onCreate*. La rularea aplicației rezultatul afișat este cel din Fig.10.

4) Selectarea informației care va fi afișate se efectuează prin utilizarea tastei programabile. Acesteia îi va corespunde un fișier ***my_menu.xml*.** Creați mai întâi un subdirector cu numele ***menu*** în directorul **layout**. Etapele necesare creării fișierului my_menu.xml sunt ilustrate în Fig.11.

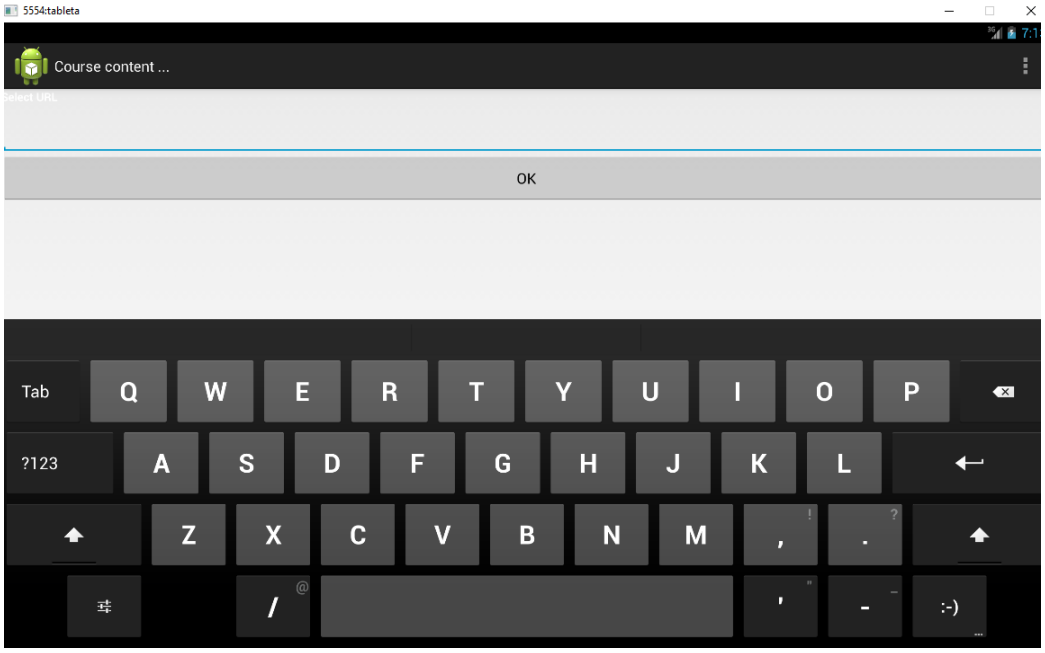


Fig.10 Interfața UI a aplicației

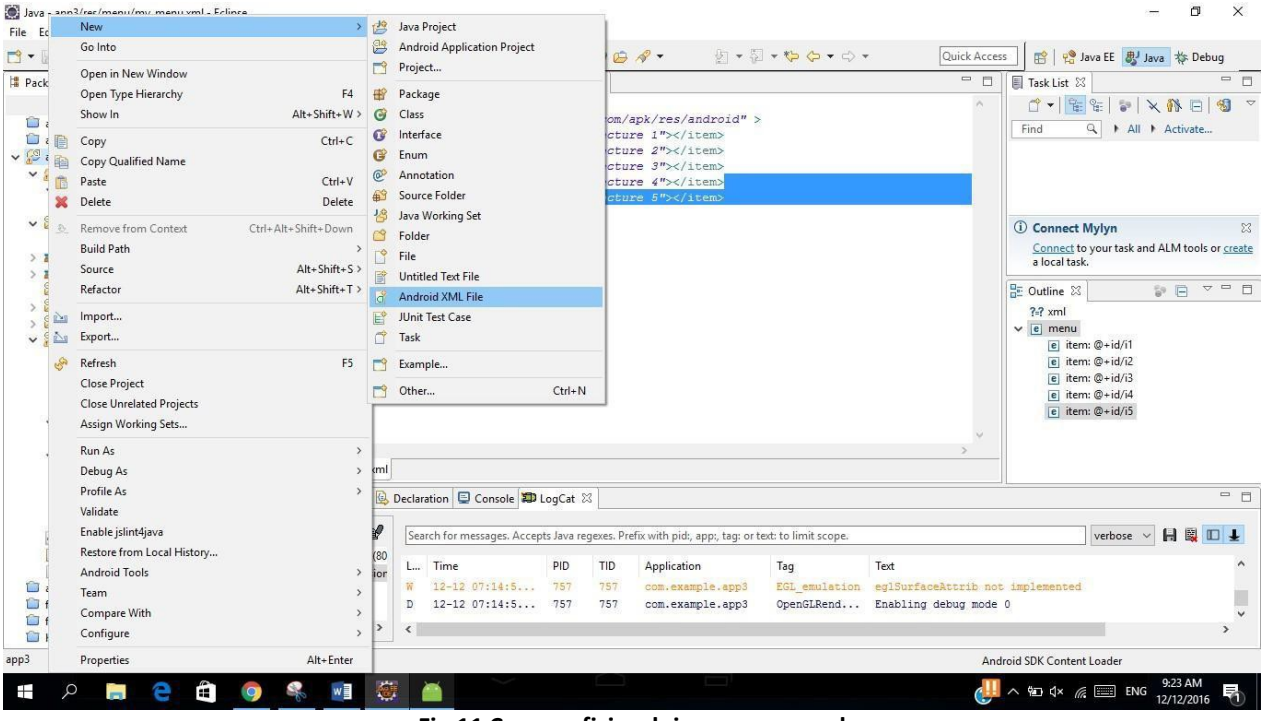


Fig.11 Crearea fișierului my_menu.xml

Conținutul fișierului trebuie editat ca în Fig.12.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"  >
 <item android:id="@+id/i1" android:title="Lecture 1"></item>
 <item android:id="@+id/i2" android:title="Lecture 2"></item>
 <item android:id="@+id/i3" android:title="Lecture 3"></item>
```

```
  <item android:id="@+id/i4" android:title="Lecture 4"></item>
  <item android:id="@+id/i5" android:title="Lecture 5"></item>
</menu>
```

Fig.12 Fișierul my_menu.xml

5) Afișarea meniului de tip pop-up presupune utilizarea următoarei secvențe de cod:

```
@Override
        public boolean onCreateOptionsMenu(Menu menu){
          MenuInflater inflater=getMenuInflater();
          inflater.inflate(R.menu.my_menu, menu);
          return true;
        }
```

Observație: pentru utilizarea în aplicație de obiecte de tip Menu următoarele clase trebuie importate în proiect:

```
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
```

În cazul în care toate elementele anterioare au fost setate corect, rezultatul este cel din Fig.13.

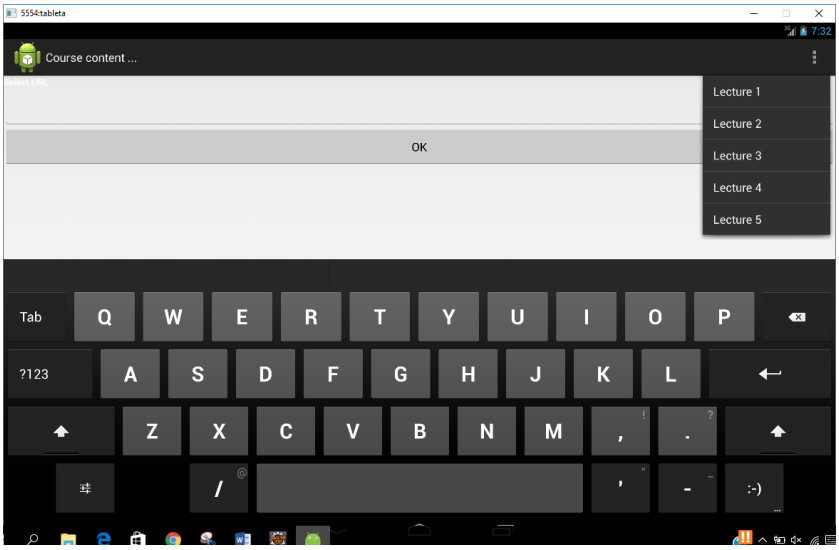


Fig.13 Intefață grafică (UI) și tastă programabilă configurată

6) Aplicația va necesita acces Internet, aceasta presupune modificarea fișierului A**ndroidManifest.xml** după cum urmează:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.example.ap
      p3" android:versionCode="1"
      android:versionName="1.0 ">

      <uses-sdk android:minSdkVersion="17"
            android:targetSdkVersion="17" />
          <uses-permission android:name="android.permission.INTERNET" />
```

```
    <application android:allowBackup="true"
        android:icon="@drawable/ic_launche r"
        android:label="@string/app_name"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Structura proiectului Android în Eclipse este cea din Fig.14.

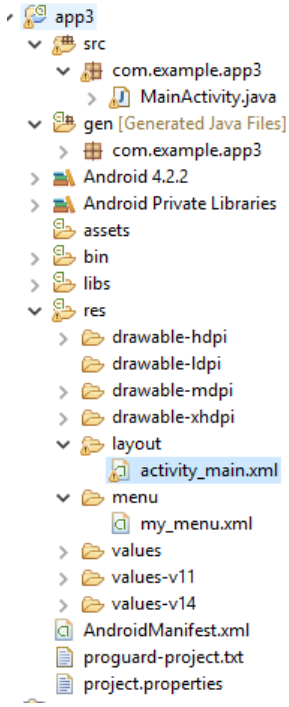


Fig.14 Proiect Android

Cod Java

Aplicatia utilizează o componentă Android de tip ***Activity***. Mai multe variabile şi metode sunt necesare pentru a face aplicația pe deplin funcţională.
Conţinutul fişierului *.java* este cel indicat în anexa 1.

5.6 Exerciții

1) Modificaţi conţinutul fişierului MainActivity.java cu conţinutul inclus în anexa 1.
2) Rulaţi aplicaţia şi observați rezultatele obținute
3) Inserați comentarii în cod care explică rolul fiecărei metode.

4) Creați un fișier de tip .doc; inserați capturi de ecran care să ilustreze funcționarea aplicației pentru diverse cursuri selectate.
5) Creați o nouă aplicație. Folosiți clase de tip *WebView* și *Spinner* care să permită obținerea unui rezultat similar cu cel obținut prin aplicația propusă în secțiunea 5.5.3. Folosiți documentația online Android pentru a comenta codul.

Bibliografie

[1] https://developer.android.com/index.html, https://www.openhandsetalliance.com/
[2] Android-Eclipse, https://www.tutorialspoint.com/android/android_eclipse.htm
[3] Android programming guide https://developer.android.com/guide/index.html

Anexa 1 Fișierul MainActivity.java

```
package com.example.first_app; import
java.io.BufferedReader; import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLConnection;

import android.app.Activity; import
android.os.Bundle; import android.os.StrictMode;
import android.view.Menu; import
android.view.MenuItem; import android.view.View;
import android.widget.Button; import
android.widget.EditText;
import android.widget.TextView;
import android.view.MenuInflater;

public class MainActivity extends Activity { String content;
TextViewtxtShow; EditText inputbox;
String address="";
        @Override
        protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (android.os.Build.VERSION.SDK_INT > 9) { StrictMode.ThreadPolicy policy =
        new StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
    }
                this.setTitle("Course content ...");
                setContentView(R.layout.activity_main); inputbox = (EditText)
                findViewById(R.id.setari); inputbox.setText("193.226.17.162");
                 txtShow = (TextView) findViewById(R.id.text); txtShow.setText("Select
                 course using the menu key"); final Button button = (Button)
                 findViewById(R.id.ok); button.setOnClickListener(new
                 Button.OnClickListener(){
                        public void onClick(View v) {
                        address=(String)inputbox.getText().toStrin
                        g(); button.setVisibility(View.GONE);
                        inputbox.setVisibility(View.GONE);
                        }
                });
        }
    @Override
    public boolean onCreateOptionsMenu(Menu menu){ MenuInflater
     inflater=getMenuInflater(); inflater.inflate(R.menu.my_menu, menu);
     return true;
    }
    private String OpenHTTPConnection(String urlString) throws IOException
     {
```

```
int response = -1;
String result="";
URL url = new URL(urlString);
URLConnection conn = url.openConnection();
try{
        HttpURLConnection httpConn = (HttpURLConnection) conn;
        HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
        BufferedReader in = new BufferedReader(new InputStreamReader( urlConn.getInputStream()));
        String inputLine;
        int lineCount = 0;
        while ((lineCount < 10) && ((inputLine = in.readLine()) != null)) {
        lineCount++;
        result += "\n" + inputLine;
        } in.close();
        urlConn.disconnect(
        );
        }
        catch (Exception ex)
        {
        throw new IOException("Eroare conexiune");
        }
        return result;
        }

private String dbInterrogation(String URL)
{
int BUFFER_SIZE =
2000; InputStream in =
null; String result=null;
try {
result= OpenHTTPConnection(URL);
} catch (IOException e1) {
// TODO Auto-generated catch block
e1.printStackTrace(); return "";
}
return result;
}
public boolean onOptionsItemSelected (MenuItem item){
        String alias_localhost="193.226.17.162";
        String url="http://"+address+"/interog.asp"; String lecture_no;
        lecture_no=txtAfis((String)item.getTitle()
        ); url=url+"?"+"lecture="+lecture_no;
```

```
            content=dbInterrogation(url);
            txtShow.setText(content); return true;
           }
        public String txtAfis(String str) {
            if (str == null) {
            return null;
            }
            StringBuffer buffer = new StringBuffer();
            char c;
            for (int i = 0; i < str.length() ; i++)
            {
            c = str.charAt(i);
            if (Character.isDigit(c))
                        {
                    buffer.append(c);
                        }
            }
            return buffer.toString();}
}
```

Lucrarea 6 Aplicații mobile folosind JME

6.1 Introducere

JME -Java Micro Edition- reprezintă o platformă software care permite dezvoltarea de aplicații pe dispozitive mobile și wireless cu capabilități limitate, telefoane mobile de tip feature phone sau dispositive embedded. Cronologic, este ultima platformă introdusă de Sun Microsystems în seria Java 2:

-**J2SE-Java 2 Standard Edition** -folosită pentru dezvoltarea de aplicații pe partea de client (desktop PC-uri, laptop-uri etc)

-**J2EE-Java 2 Enterprise Edition**-folosită pentru dezvoltarea aplicațiilor pe partea de server.

După achiziționarea companiei de Sun Microsystems de către Oracle, JME a început să fie promovată drept una dintre tehnologiile utilizate pentru dezvoltarea de aplicații pentru sisteme embedded, M2M (Machine to Machine) și IoT (Internet of Things) .

JME adresează limitările acestor tipuri de dispozitive și permite dezvoltarea de aplicații folosind concepte similare cu J2SE:

-arhitectura este aceeași: un compilator Java transpune codul Java în instrucțiuni (bytecode) pentru o mașină virtuală (abstractă) care permite rularea aplicațiilor în cod mașină;

-aplicatiile dezvoltate in JME sunt portabile: odată scrise, aplicațiile pot fi implementate și rulate pe diverse sisteme de operare, sau pe alte echipamente hardware.

Nu toate clasele J2SE sunt însă disponibile în JME: JME nu oferă suport AWT (Abstract Windowing Toolkit) pentru dezvoltarea interfețelor grafice, unele versiuni JME nu oferă suport pentru calcul în virgulă mobilă, accesul la fișiere nu este posibil decât prin pachete suplimentare opționale etc.

6.2 Configurații și profiluri JME

În definirea platformei JME, Sun Microsystems a introdus conceptul de ***configurație (configuration)***, un termen care definește un set de caracteristici software (clase Java, interfețe și API-uri, caracteristici ale mașinii Java virtuale) și hardware (capacitate memorie, rezoluție ecran, conectivitate) minimale pentru a rula o aplicație JME. Sunt definite două configurații:

-**CDC-Connected Device Configuration**. Dispozitivele tipice din această categorie (dispozitive fixe, STB-uri, PDA-uri cu acces wireless) posedă tipic următoarele caracteristici:

- procesoare pe 32 biți;
- minim 2MB RAM și minim 2MB ROM;
- conectivitate la rețea prezentă, posibil permanentă și cu lățime de bandă mare.

-**CLDC-Connection Limited Device Configuration**. Dispozitivele din această categorie sunt tipic terminale mobile de tip feature phone cu caracteristici inferioare celor din categoria CDC:

-procesoare pe 16-biți sau pe 32-biți ce operează la minim 16MHz;

-cel puțin 160 KB de memorie de tip ROM;

- minim 8KB de memorie RAM ;

-conectivitate de rețea prezentă, de obicei wireless cu lățime de bandă scăzută.

JME este definită în termeni de o arhitectură stratificată cu CDC/CLDC operând direct cu partea de hardware sau prin intermediul unui sistem de operare, oferind însă doar câteva clase de bază.

Pentru dezvoltarea de aplicații este necesară utilizarea de API-uri suplimentare. Clasele grupate în această categorie poartă numele de **profil *(profile)*.** Un profil include metode, funcții și clase suplomentare funcții neoferite de CDC/CLDC cum ar fi:

-componente UI (forme, casete text, comenzi etc);

-timere;

-conectivitate;

-persistență.

Separarea configurație- profil oferă mai multă flexibilitate pentru implementarea aplicațiilor iar JME oferă API-uri specifice la nivel de profil (de ex. funcții de tip *PIM*, conectivitate internet etc). Două profiluri principale sunt definite pentru implementarea de aplicații: ***MIDP (Mobile Information Device Profile)*** și ***PDAP (Personal Digital Assistant information Device Profile)***. Primul tip de profil este tipic utilizat pentru dezvoltarea de aplicații pentru terminale mobile de tip feature phone iar variantă a acestuia, **IMP-NG, *NG (Information Module Profile- Next Generation)*** [2], definește setul de clase pentru aplicații embedded (GPIO, I2C etc).

Pentru a asigura portabilitatea între dispozitive din aceeași clasă, producătorul trebuie să implementeze toate caracteristicile specificate pe o configurație și un profil. La dezvoltarea de aplicații pot fi folosite însă clase și API-uri suplimentare.

Exemplu: profilul **MIDP 2.0** nu oferă API-uri pentru mesagerie și funcții multimedia însă există pachetele opționale de tip **Wireless API** și **MMAPI** care implementează acest tip de facilități. Pachetele suplimentare trebuie însă să fie incluse în proiectul JME.

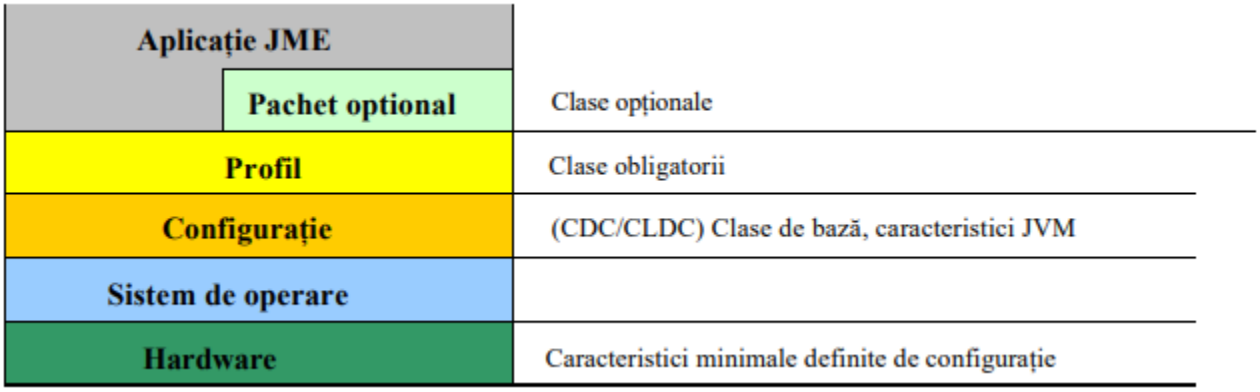


Fig.1 Aplicație generică JME

6.3 Profilul MIDP : MIDlet-uri, pachete de clase

Profilul **MIDP** este profilul cel mai utilizat și oferă funcționalitate superioară celei oferite prin CLDC.

Aplicațiile bazate pe MIDP-CLDC sunt numite **MIDlet**-uri (echivalentele **Servlet**-urilor, **Applet**-urilor din J2SE). Ca și în JSE JME utilizează termenul ***pachet (package)*** pentru a identifica o colecție de clase și interfețe. Câteva exemple de pachetele incluse în profilele MIDP 1.0 și MIDP 2.0 sunt afișate în cele ce urmează:

MIDP 1.0	
Pachet	**Descriere**
java.io	- clase și interfețe IO Java standard pentru operații IO
java.lang	- clase din J2SE (ex. Boolean, double , integer, short, string etc.)
java.util	- clase și interfețe utile din J2SE (ex: calendar, timere,
javax.microedition.io	- clase și interfețe specifice JME pentru operații IO (conectivitate la rețea)
javax.microedition.lcdui	- clase și interfețe specifice JME pentru dezvoltarea de aplicații cu interfata grafică
javax.microedition.rms	- clase și interfețe specifice JME pentru persistență
javax.microedition.midlet	- clase și interfețe pentru aplicații cu interfață grafică
MIDP 2.0	
toate cele de mai sus +	
javax.microedition.lcdui.game	- clase și interfețe pentru dezvoltarea de jocuri
javax.microedition.media	- clase și interfețe pentru aplicații multimedia
javax.microedition.media.control	- interfețe media
javax.microedition.pki	- clase și interfețe pentru conexiuni securizate

Tabel 1. Exemple de pachete incluse în profilele MIDP

6.4 Aplicația "Hello world" în JME

Cea mai simplă aplicație de tip JME MIDlet ce corespunde unei aplicații de "Hello World " este ilustrată mai jos:

J2ME code	Descriere
import javax. microedition.lcdui.*; **import javax. microedition.midlet.*;**	Import clase și interfețe pentru dezvoltarea UI import clasă abstractă MIDlet
public class HelloWorld **extends** MIDlet {	Declarare clasă; clasa Hello world va suprascrie metodele abstracte ale clasei MIDlet

public void **startApp**( ) {
 	
Display d =**Display**.**getDisplay** (this);
 	
Form f = **new Form** ("Hello world");
 	
d.**setCurrent(**f);
}
	startApp () e prima metodă care se execute la lansarea unei aplicații JME
	creează un obiect JME de tip Display afișat care va gestiona toate componentele UI care pot fi afișate pe ecranul dispozitivului;
	creează un obiect JME de tip formă care va fi afișa de obiectul de tip Display. Titlul este setat la "Hello world"

public void **destroyApp**(**boolean** force) {Î	Metodă apelată la ieșirea din aplicație
public void **pauseApp**() { }	un MIDlet poate fi întrerupt ca urmare a unui eveniment (un apel telefonic de exemplu) În această secțiune se adaugă o secvență de cod pentru revenirea MIDlet-ului în stare activă
}	**Sfârșit clasă**

6.5 Adăugarea de funcții suplimentare prin intermediul interfețelor JME

Funcționalități specifice aplicației se adaugă tipic în JME prin intermediul **interfețelor**. Fiecare pachet MIDP are propriile interfețe ([1] descriere completă); utilizarea interfețelor va fi explicată numai pentru componentele UI.

Cele mai multe aplicații J2ME de tip MIDP permit interacțiunea cu utilizatorul prin obiecte de **command**, asociate de obicei cu tasta programabilă de pe terminale de tip feature phone. Comenzile pot fi adăugate programatic iar J2ME oferă o modalitate unificată pentru a gestiona acțiunile utilizatorilor printr-o interfață standardizată numită **commandListener**. Atunci când se utilizează această interfață, o aplicație de tip MIDlet trebuie să propună o implementare pentru metoda **commandAction (Command c, Deplayable d)** definită ca prototip în interfața **commandListener**. După cum indică prototipul metodei, o singură interfață **commandListener** poate fi atașată unui obiect grafic. Exemplul următor ilustrează o versiune îmbunătățită a MIDlet-ului "Hello World" care implementează interfața **commandListener.**

Cod JME	Descriere
import javax.microedition.midlet.* ; import javax.microedition.lcdui.* ;	
public class HelloWorld extends MIDlet **implements CommandListener {** private Command cmdOne; private Form f; private Display d;	

public void startApp( ) {
d =Display.getDisplay (this);
f = new Form (“ Hello world”);
cmdOne= new Command("Quit", Command.EXIT, 0);
 	
f.addCommand(cmdOne);
d.setCurrent(f);
f.setCommandListener(this);
 	
}	

public void **destroyApp**(boolean force) { }	
public void **pauseApp**() { }	
public void commandAction (Command com, Displayable dis){ **if (dis == f){** **if (com == cmdOne){** **destroyApp(true)** **;** **notifyDestroyed(**	
}	

Rezultatul rulării aplicației este ilustrat mai jos:

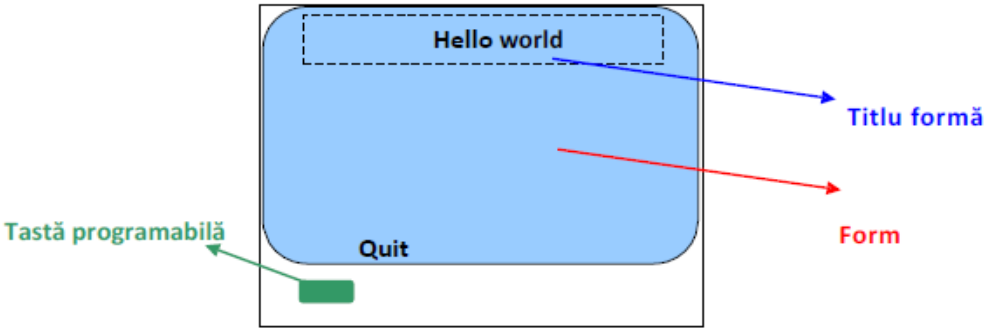


Fig.2 Aplicația Hello World și interfață commandListener

Mai multe comenzi pot fi asociate însă cu un obiect grahic. În acest caz diferitele comenzi vor fi accesibile printr-un meniu de tip pop-up.

Evenimentul initiate de utilizator pot avea loc nu doar în conjuncție cu obiecte de tip command. Elementele UI precum **ChoiceGroup** sau elemente editabile de tip text **TextField** pot fi de asemenea utilizate. Interceptarea acestor evenimente este efectuată în JME prin intermediul unui alte interfațe numite **ItemStateListener**. O aplicație de tip MIDlet care implementează o interfață **ItemStateListener** trebuie să implementeze explicit metoda **ItemStateChanged (Item i)**. Exemplul de mai jos ilustrează o posibilă utilizare a acestei inferfețe pentru aplicația Hello World O punere în aplicare practică a ItemStateListener interfață pentru "Hello World" cerere este explicat pe exemplul de mai jos:

Cod JME	**Descriere**
import javax. microedition.lcdui.* import javax. microedition. midlet.*	

public class HelloWorld extends MIDlet implements CommandListener,
ItemStateListener {
private Command cmdOne= new Command("Quit", Command.EXIT, 0);
private Form f;
private Display d;
private TextField txtInput=new TextField ("Input text", null,40, TextField.ANY);	

public void startApp( ) {
d =Display.getDisplay (this);
f = new Form (" Hello world");
f.addCommand(cmdOne); **f.append**
(txtInput); d.setCurrent(f);
f.setCommandListener(this);
f.setItemStateListener(this);
}	

public void destroyApp(boolean force) {}	
public void pauseApp) {}	

public void commandAction (Command com, Displayable dis){
if (dis == f){
if (com == cmdOne){
destroyApp(true);	

```
public void itemStateChanged (Item i){
if (i==txtInput)
      {
this.f.setTitle(txtInput.getString());
      }
}
}
```

6. 6 Dezvoltarea aplicațiilor JME

Aplicațiile JME sunt dezvoltate folosind o platformă de tip SDK JME, disponibilă în pagina de web a companiei Oracle. Automatizarea procesului de creare a aplicațiilor presupune și utilizarea unui IDE, mediul de dezvoltare oficial fiind Netbeans. Acesta permite crearea aplicațiilor în format *JAR (packaged Java executable)*.

Exerciții

1. Folosiți mediul de dezvoltare Netbeans 7.4 pentru a crea aplicația Hello World din secțiunea 6.4. (*New project* în meniul *File, Mobile->Mobile application,* nume proiect HelloWorld, deselectați opțiunea "Create Default Package and Main Executable Class", setați drept parametri de configurare CLDC 1.0, MIDP 2.1 și emulatorul implicit.
2. În proiectul creat adăugați codul din secțiunea 6.4
3. Modificați proiectul adăugând interfețele mai sus menționate și codul asociat. Comentați codul și analizați rezultatul pe simulatorul JME.

6.7 Aplicația Hello World în Netbeans

NetBeans 7.4 IDE permite automatizarea procesului de creare a clasei și a structurii aferente de directoare a aplicației "Hello World", oferind de asemenea facilități ce permit ușurarea procesului de creare a interfețelor grafice prin utilizarea opțiunii "Create Default Package and Main Executable Class" disponibilă la crearea unei noi aplicații. Folosirea acestei facilități permite editarea în mod grafic a interfeței cu utilizatorul (fig. 4) prin intermediul unor widget-uri structurate conform pachetului **javax.microedition.lcdui.** Codul generat poate fi inspectat accesând tab-ul *Source.*

Exerciții

1. Folosiți mediul de dezvoltare Netbeans 7.4 pentru a crea aplicația Hello World din secțiunea 6.4. (*New project* în meniul *File, Mobile->Mobile application,* nume proiect HelloWorld, **selectând de această dată** opțiunea "Create Default Package and Main Executable Class", setați drept parametri de configurare CLDC 1.0, MIDP 2.1 și emulatorul implicit.
2. Analizați secțiunea de cod creată automat și evidențiați diferențele față de versiunea creată manual.

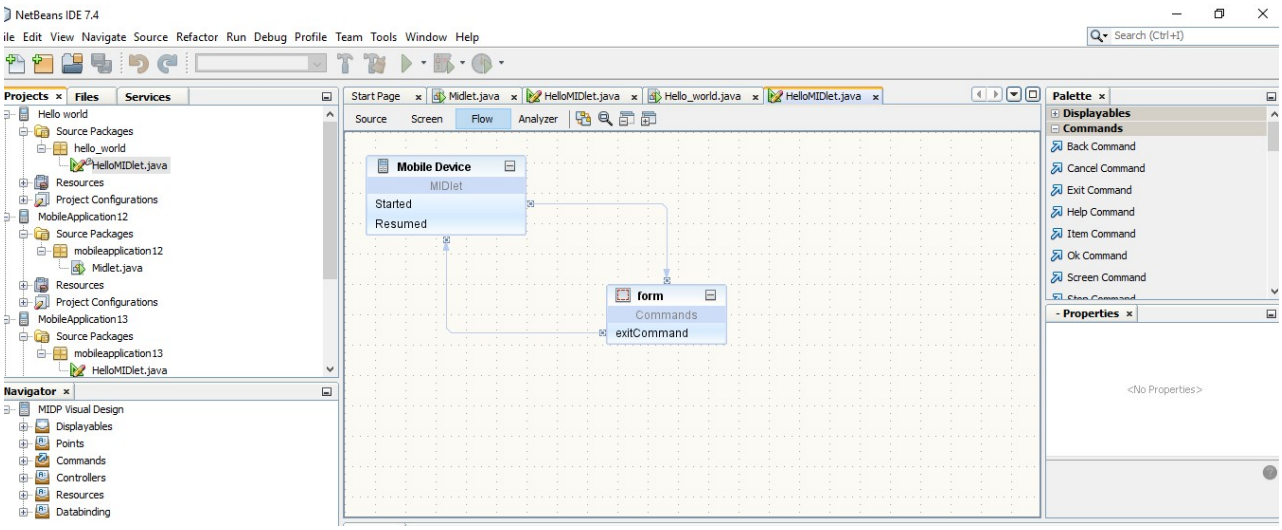


Fig.3 Aplicația Hello World creată automat folosind NetBeans 7.4

6.8 Suport pentru comunicații și acces Internet în JME

JME oferă prin CLDC suport pentru comunicații și acces Internet. Acesta constă într-o clasă (***Conector*****)** si 7 interfețe interfețe ***Connection, ContentConnection, DatagramConnection, InputConnection, OutputConnection, StreamConnection,*** **and** ***StreamConnectionNotifier***). Acestea sunt incluse pachetul **javax.microedition.io**. Interfețele permit implementarea de mai multe moduri de transmiune: comunicații seriale, comunicații cu comutație de pachete în mod datagramă, socket-uri, conexiuni HTTP cu un web server etc.

Pentru deschiderea unei conexiuni de tip HTTP, metoda folosită este
Conector. Open (String Connect);
Sirul transmis ca parametru conține un URL-ul în format standard.

MIDP oferă de asemenea o clasă ***HTTPConnection*** pentru stabilirea conexiunilor de acest tip. Cele mai importante metode implementate de această aclasă sunt ***DataInputStream openDataInputStream (); DataOutputStream openDataOutputStream();*****.** Obiectele de tip input stream sunt folosite pentru a citi răspunsurile transmise de serverul web.

6.9 Exemple de aplicații JME

6.9.1 Interogarea unei baze de date plasate pe un server de web

Aplicația următoare permite afișarea conținutului dynamic al unei baze de date pe un dispozitiv compatibil JME, de o manieră similară cu cea ilustrată în lucrările de laborator dedicate aplicațiilor web și, respectiv Android.

Etapele dezvoltării aplicației sunt următoarele:
1. Accesați *New Project->Mobile-> Mobile application*
2. Setați numele proiectului *WebApp*
3. Debifați opțiunea ““Create Default Package and Main Executable Class” -> *Next*

4 Setati configurările : *Default Color Phone, CLDC 1.1, MIDP 2.0 -> Finish*
5. Accesați pachetul creat de IDE și selectați *New-> Visual Midlet*
6. Setați numele aplicației to *WebApplication*

7. Adăugați proiectului un obiect de tip **Form** și setați-I numele *frmLectures* (widget-urile *Flow* și *Screen*)
8. Adăugați un obiect de tip **Command** pe forma *frmLectures* și setați proprietatea *Label=Exit*
9. Repetați pașii anteriori pentru a adăuga un nou obiect de tip **Command**; setați proprietatea *Label=Retreive data*. Modificați numele acestuia în *cmdGoToURL*
10. Adăugați un element de tip **ChoiceGroup** formei *frmLectures* și setați proprietățile *Label= MCS lectures*. Selectați tipul POPUP.
11. Adaugați 4 obiecte e tip **Choice Element** și setați proprietățile *String* asociate la *Lecture 1, Lecture 2, Lecture 3* și *Lecture 4*.
12. Adăugați un element de tip **TextField** formei și setați proprietatea *Label=Content*.

Interfața cu utilizatorul astfel creată are următorul aspect:

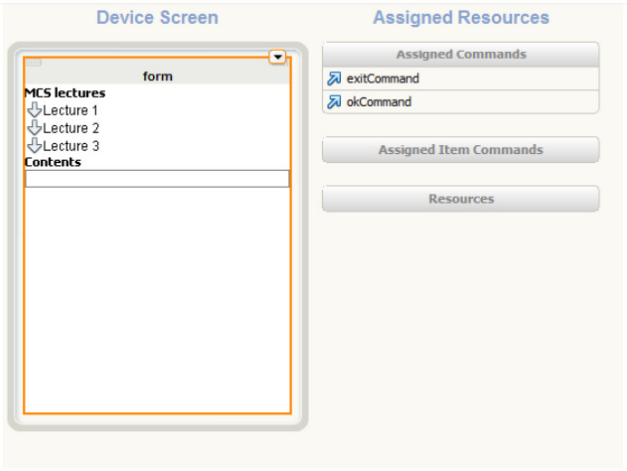
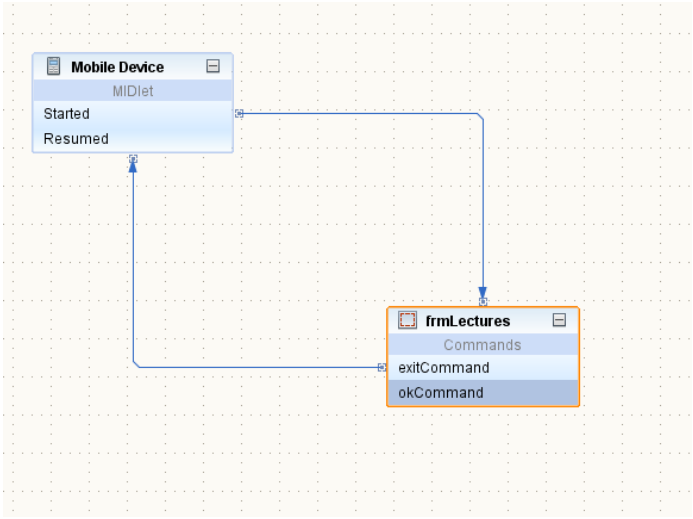


Fig.4 Forma frmLectures

13. Folosind widget-ul *Flow* , conectați elementele UI după cum urmează:



14. Adăugați o metodă numită ***testASP***(). Modificați corpul metodei după cum urmează:

```
public String testASP() throws IOException {
    StringBuffer messagebuffer = new StringBuffer(1200);
    try {
      String uri = "http://193.226.17.162/interog.asp?lecture=";

      int i = 0;

      int l_no = choiceGroup.getSelectedIndex();

      uri = uri + (l_no + 1);

      HttpConnection conn = (HttpConnection) Connector.open(uri);
      DataInputStream dis = new DataInputStream(conn.openInputStream());
      int ch;

      long len = conn.getLength();
      String readcharacter;
      for (i = 0; i < len; i++) {
        if ((ch = dis.read()) != -1) {
          messagebuffer.append((char) ch);
        } else {
          while ((ch = dis.read()) != -1)
      //până la sfârșitul mesajului returnat
          {
          messagebuffer.append((char) ch);
          }
        }
      }

      dis.close();
```

```
    } catch (ConnectionNotFoundException e) {
      System.out.println("Eroare conexiune");
    }

    return messagebuffer.toString();

  }
```

15. Modificați aplicația astfel încât să implementeze interfața **Runnable**.:

```
public class WebApplication extends MIDlet implements
javax.microedition.lcdui.CommandListener, Runnable {….
```

16. Modificați corpul metodei **run** asociate interfeței după cum urmează:

```
public void run() {
    try {
       textField.setString(testASP());
       System.out.println(testASP());
    } catch (IOException e) {
       System.out.println("Eroare conexiun");
    }
```

17. Adăugați secvența de cod următoare în secțiunea **Post-Action User Code** a obiectului *cmdGoToURL*:

```
Thread t = new Thread(this);
        t.start();
```

Exerciții

1. Lansați în execuție aplicația și notați rezultatele obținute.
2. Inspectați codul propus spre implementare; inserați, folosind documentația JME comentarii care să ilustreze rolul fiecărei clase și metode.

6.9.2 Aplicație de tip convertor valutar JME.

Aplicația este plasată pe pagina de web a disciplinei. Descărcați-o și importați-o în Netbeans.

Exerciții

1. Analizați modul de funcționare al aplicației (widget-urile ***Flow*** și ***Screen****)*
2. Cum este implementat mecanismul de persistență?
3. Care ar fi efectul neimplementării interfeței **ItemStateListener**
4. Adăugați toate elementele necesare pentru a permite conversia dintr-o/într-o altă monedă, cu un curs editabil.

Bibliografie

[1] MID profile - https://docs.oracle.com/javame/config/cldc/ref-impl/midp2.0/jsr118/index.html
[2] IMP NG profile - https://jcp.org/en/jsr/detail?id=228
[3] Oracle® Java Micro Edition Software Development Kit Developer's Guide - https://docs.oracle.com/javame/dev-tools/jme-sdk-3.2/ecl/html/runsamples.htm#Z40003f61316245

Lucrarea 7 Rețele LTE. Arhitectura, proceduri pe interfața radio

7.1 Introducere

Fig. 1 prezintă arhitectura generică a unui sistem **EPS (Evolved Packet System)**, tehnologie 3GPP care succede sistemelor de generația 3-a (UMTS).

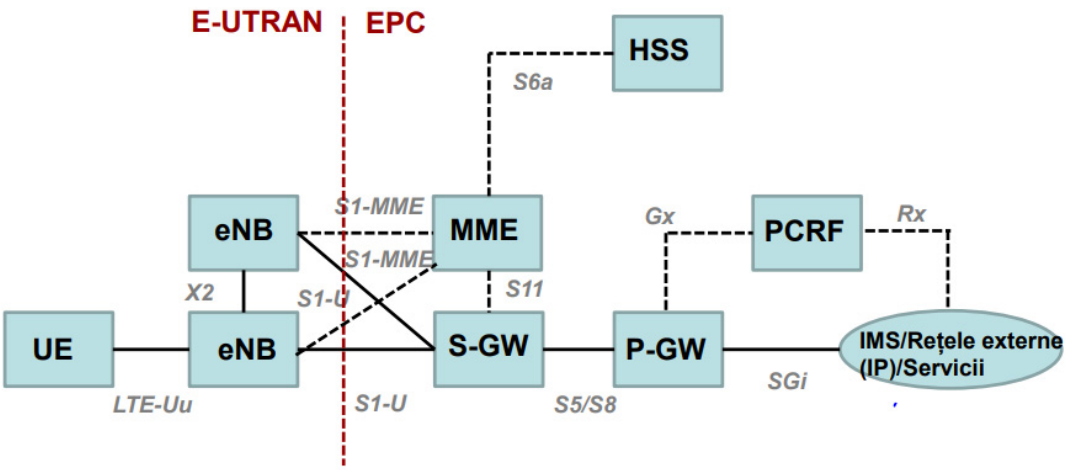


Fig.1 Arhitectura unui sistem de comunicații mobile de generație 4G EPS(Evolved Packet Systm)

Introducerea **EPS** ca evoluție a sistemelor de generația a 3-a a fost prevăzută în anul 2004 ca evoluție pe termen lung a sistemelor UMTS. În acest context LTE **(Long Term Evolution)**, numele unui proiect care a urmărit creșterea ratelor de transfer pe interfața radio și reducerea întârzierilor la transmiterea pachetelor, a debutat în ianuarie 2005 ca un studiu de fezabilitate, finalizat printr-un raport tehnic (3GPP TR 25.912). Sistemele EPS (denumire comercială LTE) sunt considerate în prezent a fi al patrulea pas major (sau 4G) în evoluția rețelelor mobile.

Similar sistemelor de generație anterioară, arhitectura pune în evidență următoarele componente:

- **UE (User Equipment)**- stații mobile
- **E-UTRAN (Evolved Universal Terestrial Acces Network)** – rețea de acces
- **EPC (Evolved Packet Core)** – rețea nucleu

Rețelele LTE permit interoperabilitatea cu rețele de generație anterioară. Specificațiile LTE adresează atât echipamentele din E-UTRAN și EPC cât și stațiile mobile UE. Tabelul de mai jos ilustrează categoriile de stații mobile indicate în specificațiile inițiale 3GPP Release 8 (3GPP TS 36.306).

Categoria UE trebuie comunicată în etapa de stabilire a unei conexiuni.

LTE specifică OFDMA și SC-FDMA ca tehnici de acces multiplu pentru sensurile de convorbire downlink și respectiv uplink. Pentru acest tip de tehnici derivate din OFDM alocarea resurselor este într-un plan timp frecvență sub controlui unui echipament de tip **eNB**

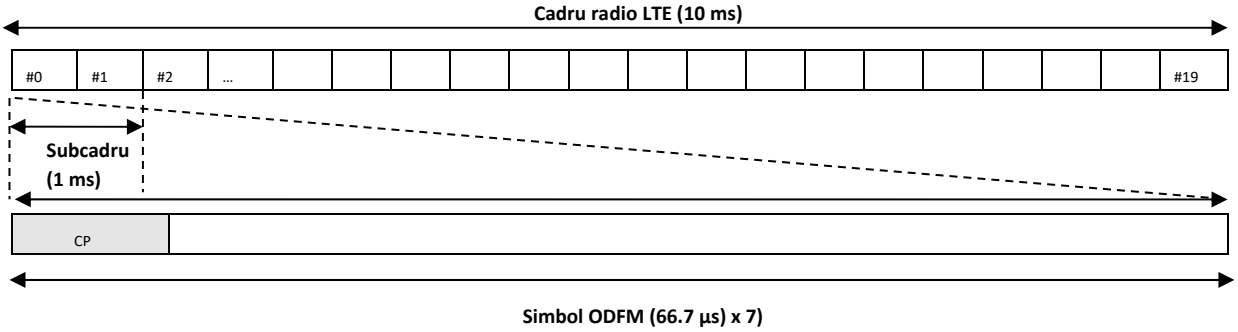
(evolved Node B), pe baza unei structuri recurente numite generic **cadru radio LTE**.

Categorie UE	Debit maxim (Mbiți/s)		Modulație		Număr antene de recepție	Număr maxim de straturi
	Downlink	Uplink	Downlink	Uplink		
1	10	5	QPSK, 16QAM, 64QAM	QPSK, 16QAM	2	1
2	50	25			2	2
3	100	50			2	2
4	150	50			2	2
5	300	75		QPSK, 16QAM, 64QAM	4	4

7.2 Transmisii în stratul fizic LTE

În domeniul timp, intervale temporale LTE sunt exprimate ca multiplii ai perioadei de simbol $T_b = 1/30720000$. Cadrul radio LTE are o durată de 10 MS ($T_{cadru} = 307200 \cdot T_b$). Fiecare cadru este împărțit în 10 subcadre de durată de 1 ms ($T_{subcadru} = 30720 \cdot T_b$).

Alocarea de resurse OFDMA sau SC-FDMA în timp se face cu rezoluția dată de un subcadru, atât pentru downlink și uplink. Fiecare subcadru este format din două sloturi de durată 0.5 ms ($T_{slot} = 15360 \cdot T_S$). Fiecare slot poate transporta un număr de simboluri OFDM care pot fi fie 7 (prefix ciclic CP normal) sau 6 (prefix ciclic extins). Durata unui simbol este definită prin $T_s = 2048 \cdot T_b = 66.7$ µs (=1/15kHz – ultimul termen definește ecartul între purtătoarele OFDM în domeniul frecvență).

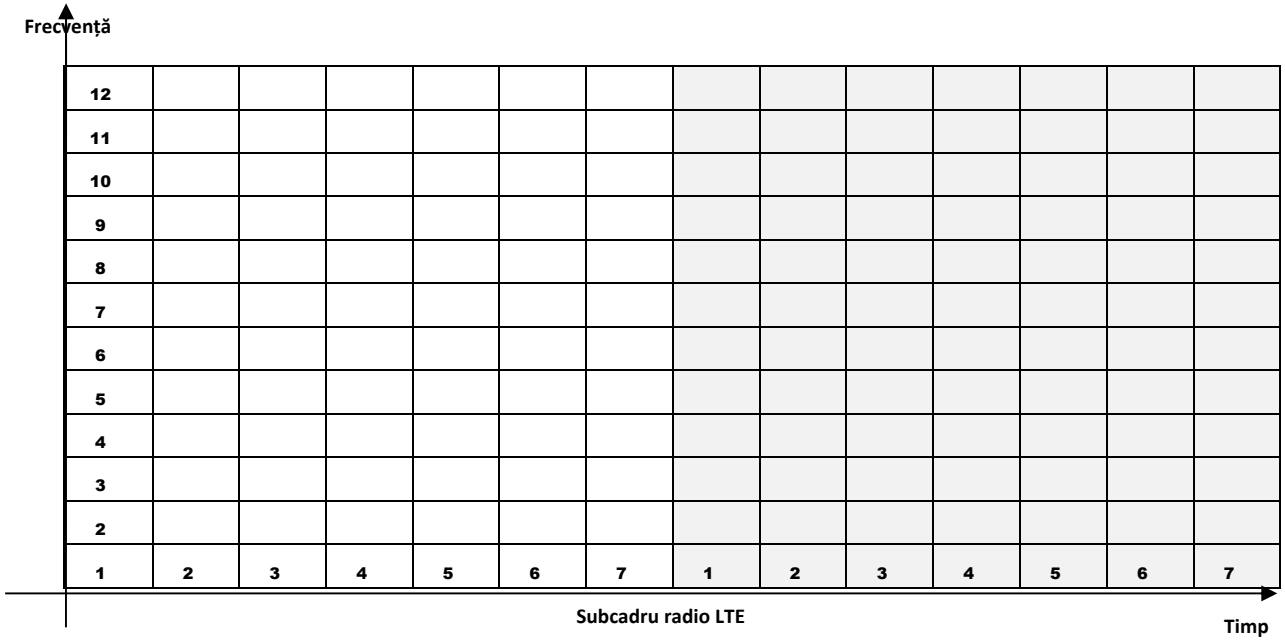


Durata asociată prefixului ciclic a fost aleasă în așa fel încât să fie mai mare decât parametrii statistici de tip delay spread care caracterizează împrăștierea temporală pe canale de propagare mobile pentru a evita ISI.

În domeniul de frecvență, numărul de subpurtătoare N variază de la 128 la 2048, în funcție de lățime de bandă scalabilă permisă în LTE (valori tipice 512/ 1024 pentru lărgimi de bandă de 5 /10 MHz).

Alocarea de resurse se face în utilizând blocuri de resurse elementare (**Resource Block**)

definite în planul timp frecvență de un număr de 12 purtătoare și de 7 simboluri OFDM (pentru operare cu un prefix ciclic normal) pe durata unui subcadru radio LTE.



Pentru un subcadru un set de resurse poate fi asignat unui singur utilizator, sarcina asignării blocurilor de resurse fiind asignată stratului MAC din eNB.

Numărul de blocuri de resurse disponibil depinde de lărgimea de bandă setată într=o celulă. Tabelul următor rezumă caracteristicile principale pentru diverse configurații de celule LTE.

Lărgime de bandă [MHz]	1.4	3	5	10	20
Număr de puncte FFT	128	256	512	1024	2048
Număr RB/slot	6	15	25	50	100
Număr de subpurtătoare utilizate	72	180	300	600	1200

Blocurile de resurse sunt utilizate pentru a transporta atât date utilizator cât și semnalizări și semnale de referință.

Figura 2 ilustrează structura unui cadru downlink LTE în mod de operare FDD pentru următorii parametrii: lărgime de bandă 1.4 MHz, prefix ciclic normal, numărul de porturi de antenă 2, număr de simboluri OFDM pentru canale de control de tip PDCCH și PHICH din fiecare subcadru.

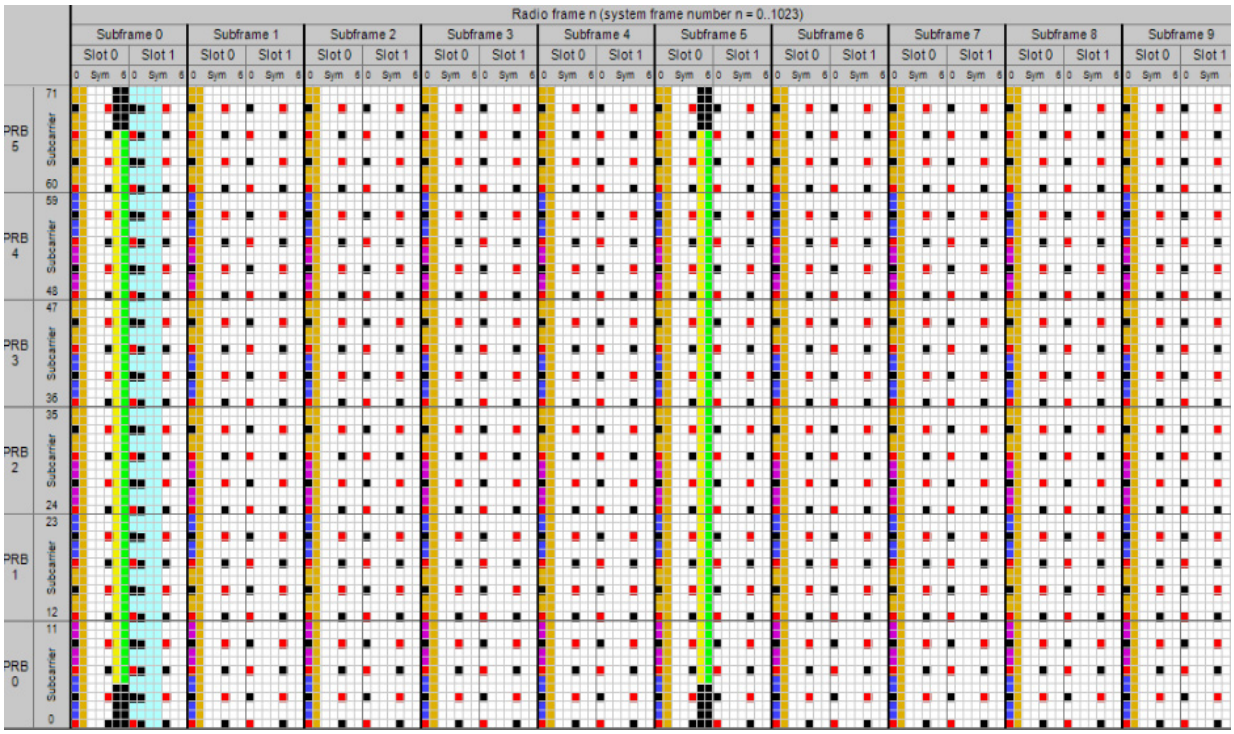


Fig.2 Structura unui cadru LTE downlink (http://niviuk.free.fr/lte_resource_grid.html**)**

PSS - Primary Synchronization Sequence
SSS - Secondary Syncronization Sequence
PBCH – Physical Broadcast Channel
RS -Reference Signal
PCHICH- Pyhiscal Control Format Indicator Channel
PHICH- Phyiscal Hybrid ARQ Indicator Channel
PDCCH – Physical Downlink Control Channel
Resurse disponibile pentru PDSCH (Physical Downlink Shared Channel).

7.3 Proceduri pe interfața radio

7.3.1 Procedura de selecție a unei celule

Funcție de capabilitățile tehnice UE scanează toate canalele posibile pentru operare LTE (eARFCn – în specificațiile LTE, valori posibile 0-65535) în toate benzile de frecvențe pe care le implementează. Pentru fiecare canal UE întreține o listă ordonată pe bază de măsurători de tip RSSI. Pentru fiecare canal detectat ca având o valoare RSSI mai mare decât un anumit prag UE se sincronizează la nivel de cadru radio prin detectarea PSS și SSS.

Secvențele de sincronizare sunt transmise de două ori într-un cadru radio LTE fiind mapate pe blocurile centrale de resurse în domeniul frecvență, independent de lățimea de bandă setată din celulă. În domeniul timp acestea sunt transmise de asemenea pe poziții fixe.

PSS – sunt secvențe de tip Zadoff-Chu ce prezintă foarte bune proprietăți de autocorelație. Detecția unei anumite secvențe (din 3 valori posibile P_k) pe bază de autocorelație, permite stației mobile să se sincronizeze la nivel de slot cu transmisiile unui anumit eNB. Detecția PSS permite

sincronizarea la nivel de slot.

SSS- sunt secvențe de 62 de simboluri formate din concatenarea a două secvențe de 31 de simboluri. Acestea sunt mapate diferite în subcadrele 0 și 5. Standarul definește 168 de posibilități pentru acest tip de secvențe (S_j) iar detecția de face de asemenea prin autocorelație. Detecția SSS permite sincronizarea la nivel de cadru.

Cei doi parametri detectați permit stațiilor mobile să se sincronizeze la nivel de cadru radio cu o anumită celulă și să determine un identificator specific acestuia (**Physical Cell ID= $3xS_j+P_k$**), cu valori posibile între 0 și 503.

7.3.2 Citire informații difuzate (MIB și SIB)

Procedura se efectuează pentru fiecare canal LTE detectat în etapa precedentă ca având un nivel RSSI superior pragului și presupune citirea informațiilor trimise pe PBCH (mapat de asemenea pe 12 purtătoare/ 6RB-uri, independent de lărgimea de bandă setată în celulă) și, mai apoi, pe PDSCH.

Informatiile difuzate în LTE sunt, ca și UMTS, structurate ierarhic, după cum urmează:

- **MIB (Master Information Block)** – include informații statice transmise cu o periodicitate de 40 de ms. Informațiile transmise includ informații necesare stratului fizic: lărgime de bandă, **SFN -System Frame Number –** parametru LTE prin care se numerotează ciclic cadrele radio LTE, modul de **configurare PHICH**, **numărul de antene de transmisie** la nivel eNB, informații privind modul de transmisete a celorlalte informații difuzate. Informațiile transmise permit UE și să determine pozițiile simbolurilor de referință din planul timp frecvență pentru o sincronizare mai precisă. PBCH se transmite folosind tehnici de diversiate multiantenă de tip SFBC.
- **System Information Block Type 1 –** trimis cu o periodicitate de 80 ms pe PDSCH, include informații privind modul de transmitere al celorlalte informații difuzate dinamice (alte SIB-uri), informații privind operatorul asociat celulei (MNC, MCC), codul ariei de urmărire (TAC) etc.
- **System Information Block 2,3… –** standardul prevede mai multe tipuri de mesaje de acest tip, parte din acestea fiind opționale. Unul dintre aceste mesaje este **System Information Block Type 2** care include informații legate de configurarea canalelor comune ce permit atașarea ulterioară la rețea.

Întrebări

1. *Folosiți aplicația dedicată pentru analiza mesajelor de semnalizare LTE instalată pe calculatoarele din laborator și deschideți fișierul **Proceduri_idle_conectare_1.isf.** Reprezentați o diagramă simplicată de indicând mesajele de semnalizare ce includ informațiile difuzate transmise în direcție downlink pentru procedura de selecție a celulei. Folosind adresa: http://www.rfwireless-world.com/Terminology/LTE-MIB-SIB-system-information-blocks.html și specificația 3GPP relevantă (3GPP TS 25.331) identificați și identificați principalele tipuri de informații trimise.*
2. *Repetați punctul anterior pentru fișierul **Proceduri_idle_conectare_2.isf** ce include și mesaje de semnalizare în stare inactive capturate pentru platforma din laborator.*

3. *Folosiți facilitățile oferite de aplicația web disponibilă la adresa http://niviuk.free.fr/lte_resource_grid.html pentru a identifica și vizualiza modul de transmisie în direcție downlink a cadrelor radio LTE.*

7.3.3 Procedura de stabilire a unei conexiuni RRC

Similar UMTS, pentru un dialog cu rețeaua nucleu, o stație mobile trebuie să solicite stabilirea unei conexiuni RRC. Aceasta are ca rezultat alocarea unui serviciu de transport semnalizări (***SRB- Signaling Radio Bearer***) ca suport pentru schimbul de mesaje între UE și eNB. LTE definește doar 3 tipuri de SRB-uri:

- SRB0 este pentru mesajele RRC folosind canale comune;
- SRB1 este pentru mesaje RRC pe DCHH (pot include un mesaj NAS), precum și pentru mesajele NAS înainte de stabilirea de SRB2.
- SRB2 este pentru mesaje RRC care includ rapoarte cu măsurători sau mesajele NAS, toate folosind DCCH canal logic. SRB2 are o prioritate mai mică decât SRB1 și este întotdeauna configurat de eNB doar după procedurile de securitate.

Schimbul de mesaje este similar celui definit în specificațiile UTMS:

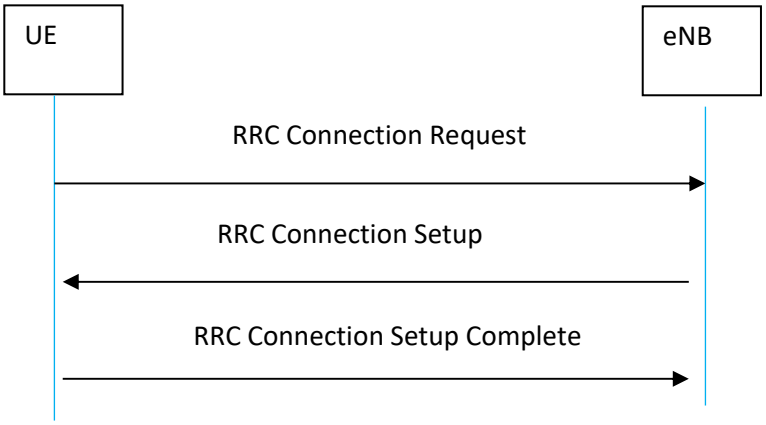


Fig.3 Stabilirea unei conexiuni cu E-UTRAN

O astfel de procedură configurează complet stiva de protocoale RLC/MAC/Strat fizic. Cererea de conectare este transmisă pe canale comune folosind o procedură de acces aleatory care are ca drept rezultat asignarea unui identificator de tip (T)C-RNTI - (Temporary) Cell Radio Network Temporary Identity. Acesta este folosit ulterior de stratul MAC pentru asignarea de resurse pe PDCCH downlink cu o rezoluție de un subcadru radio prin mesaje de control a stratului de fizic de tip **DCI (Downlink Control Information)**.

Procedura are drept rezultat și setarea parametrilor pentru transmisii uplink (**PUCCH**).

Întrebări

1. *Folosiți fișierele de tip log din secțiunea anterioară a indica mesajele de semnalizare și parametrii setați ca rezultat al unei astfel de procedure.*
2. *Ambele proceduri fac parte din scenarii semnalizare mai complexe. Folosind explicațiile de la curs indicați care sunt acestea și, respectiv, care este rolul fiecărui mesaj de semnalizare.*

7.4 Schema bloc a platformei de laborator

Platforma hardware și software din laborator are următoarea arhitectură simplificată[2].

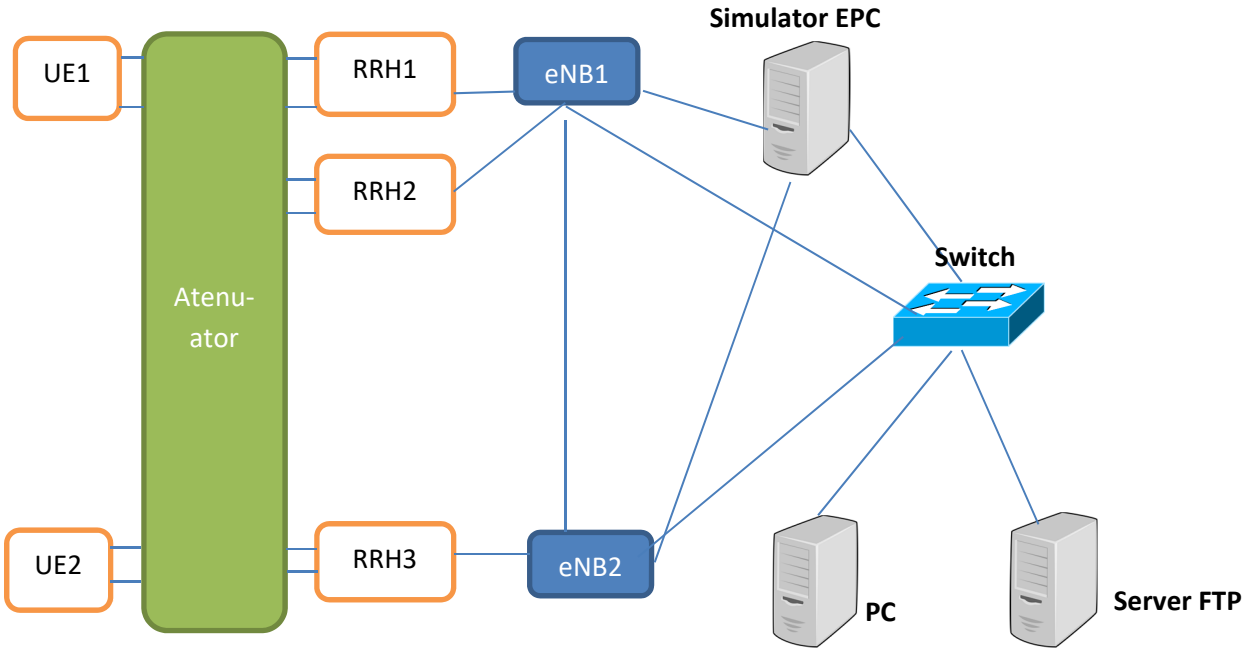


Fig.4 Schema bloc simplificată a platformei de laborator

Funcțiile echipamentelor incluse în EPC sunt îndeplinite de echipamentul **Simulator EPC** în timp ce rețeaua de acces este echipată cu 2 eNB-uri conectate la 3 echipamente de tip **RRH (Remote Radio Head)**, componente de RF speciale plasate în imediata vecinătate a sistemului de antene care include partea de procesare analogică a unui eNB și funcții de conversie din semnale optice în semnale electrice și invers.

Arhitectura de interconectare pentru eNB1 permite configurarea în mod de operare **Carrier Aggregation** cu folosirea în aceeași celulă a două purtătoare separate în configurație multiantenă de tip 2x2 MIMO.

[2] Dotarea laboratorului de Comunicații mobile din cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a UTCN, cu echipamente 4G a fost realizată cu sprijinul companiei Nokia România..

Fig.5 Echipamente de tip RRH

Echipamentele de tip eNB pot fi setate în diferite configurații (lărgime de bandă, cu agregarea purtătoarelor sau nu, folosind fișiere de configurare specifice).

Întrebări

1. *Presupunând că stațiile de bază nu sunt înregistrate în rețea, indicați care este secvența de mesaje de semnalizare necesară pentru permite un transfer downlink către una dintre stațiile mobile și cum se mapează aceasta pe comenzi inițiate asupra diverselor componente ale platformei.*
2. *Folosind utilitarele dedicate notați principalele mesaje de semnalizare rezultate ca urmare a inițierii secvenței de comenzi.*

Bibliografie

[1] 3GPP TR 25.912 - Feasibility study for evolved Universal Terrestrial Radio Access (UTRA) and Universal Terrestrial Radio Access Network (UTRAN)

[2] 3GPP TS 36.306 - Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio access capabilities

[3] 3GPP TS 23.002 Network Architecture

[4] Demo Rack User Guide- Nokia

[5] Comunicații mobile, notițe de curs, http://ares.utcluj.ro