Protokoll technológia vizsga felkészülés (korábbi vizsgák alapján)

ZH –ig TARTÓ ANYAGRÉSZES FELADATOK

Európai / Amerikai PCM

ISDN PRA (Primary Rate Access) az alközpontok illesztése a nyilvános hálózathoz
2 verzió terjedt el, Európai és az Amerikai. Az Európai jobb minőségű, de ára van.
Európai felépítése: 30+2 csatornás PCM (Pulse Code Modulation), ahol 30 adatcsatorna és 2 jelzés csatorna, minden időrés 8 bites. 1 keret 125 mikrosec hosszú, ezt felosztjuk 32 egyforma részre, melyet időrésnek hívunk (Channel Time Slot). Minden keretben a 0. és 16. időrésben jelzés információt viszünk át (signaling).

0	1	..	..	15	16	17	..	..	31

1 PCM csatorna 64 Kbps sávszélességű, ebből 32 db, így 32*64 Kbps = 2048 Kbps = 2.048 Mbps

16 db ilyen keretet (Frame) összefogva egy MultiKeretet kapunk (MultiFrame), amelynek a hossza (időtartama) 2 ms, mivel 16*125 us = 2000 us = 2 ms. Egy ilyen multikeret együttesen látja el a csatornák vezérlését, mivel keretenként 2 csatornára vonatkozó jelzést tudunk átvenni, így 16 kerettel 32 csatornára vonatkozó jelzést tudunk átvinni, és így az összes, 32 csatornához tartozó jelzést átvittük.

Az Európai szabvány még bithiba jelzésre is alkalmas egy 4 bites CRC-vel (Cyclic Redundancy Code), amelyet a Multikeret felére, azaz 8 db keretre generálunk (Sub-Multiframe → a multikeret 2*8 alkeretre van bontva). Ennek nem a beszédjel javítása a cél, hanem a vonal minőségének a jelzése, ellenőrzése, mivel ha sokat kell javítani, akkor rossz a vonal minősége, ha pedig hibátlanul átjön, akkor megfelelő.

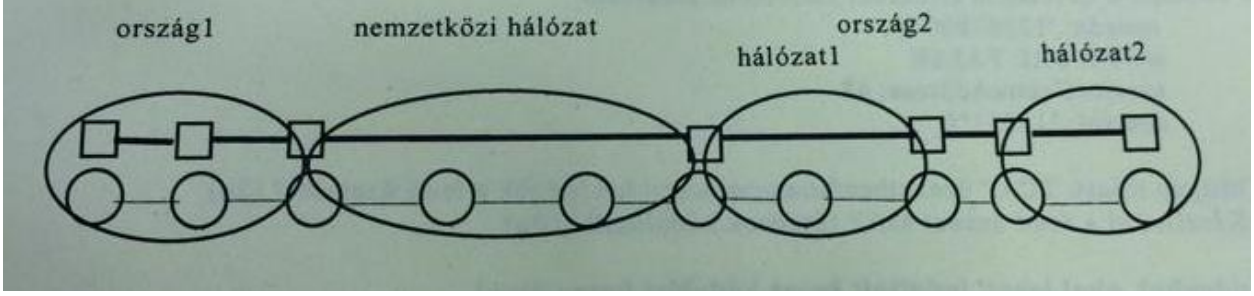
Az Amerikai az európai szabványnál egyszerűbb. 24 csatornás felépítésű, mindegyik csatorna beszéd csatorna, így nincs dedikált jelzésátviteli csatorna. 1 keret itt is 125 us hosszú, ezt osztották fel 24 egyenlő részre. 1 Multikeret 12 keretből áll, így 12*125 us = 1500 us = 1.5 ms hosszú.
Sávszélessége 24*64 Kbps = 1536 Kbps = 1.536 Mbps. 1 keret hossza 193 bit, amely 192 bit adat + 1 bit jelzés, amely szinkronkódszó.
A jelzésátvitel úgy történik, hogy minden 6. és 12. keretben “lecsípik” az utolsó hang bitet, és itt visznek át jelzéseket. Ezért minden 6. mintának a pontossága fele akkora, mint a többié. Beszélgetésnél ez nem probléma, mivel az emberi fül gyakorlatilag nem veszi észre, de az adatátvitel ilyen formában megvalósíthatatlan → Megoldás adatátvitelre: minden 6. csatornát ki kell hagyni, így megoldható, de nem hatékony, rossz kihasználtságu.

Ismertesse milyen lépésekben történik a jelzéshálózat átkonfigurálása, ha ezek a linkek meghibásodnak típusú feladat

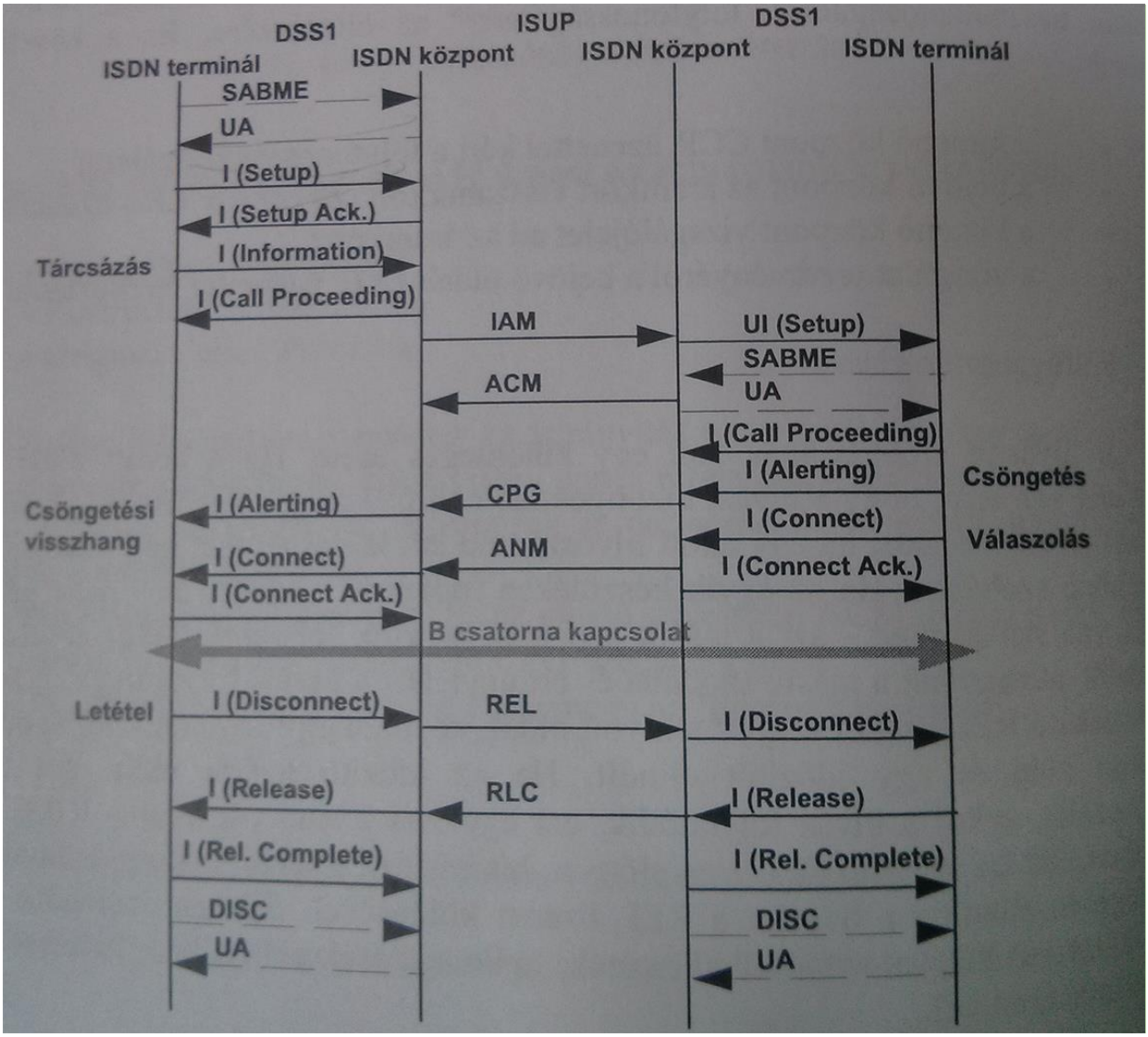
Hatos minta hálózatos feladat. Zh témakör.

Hívás felépítős feladat sok központon

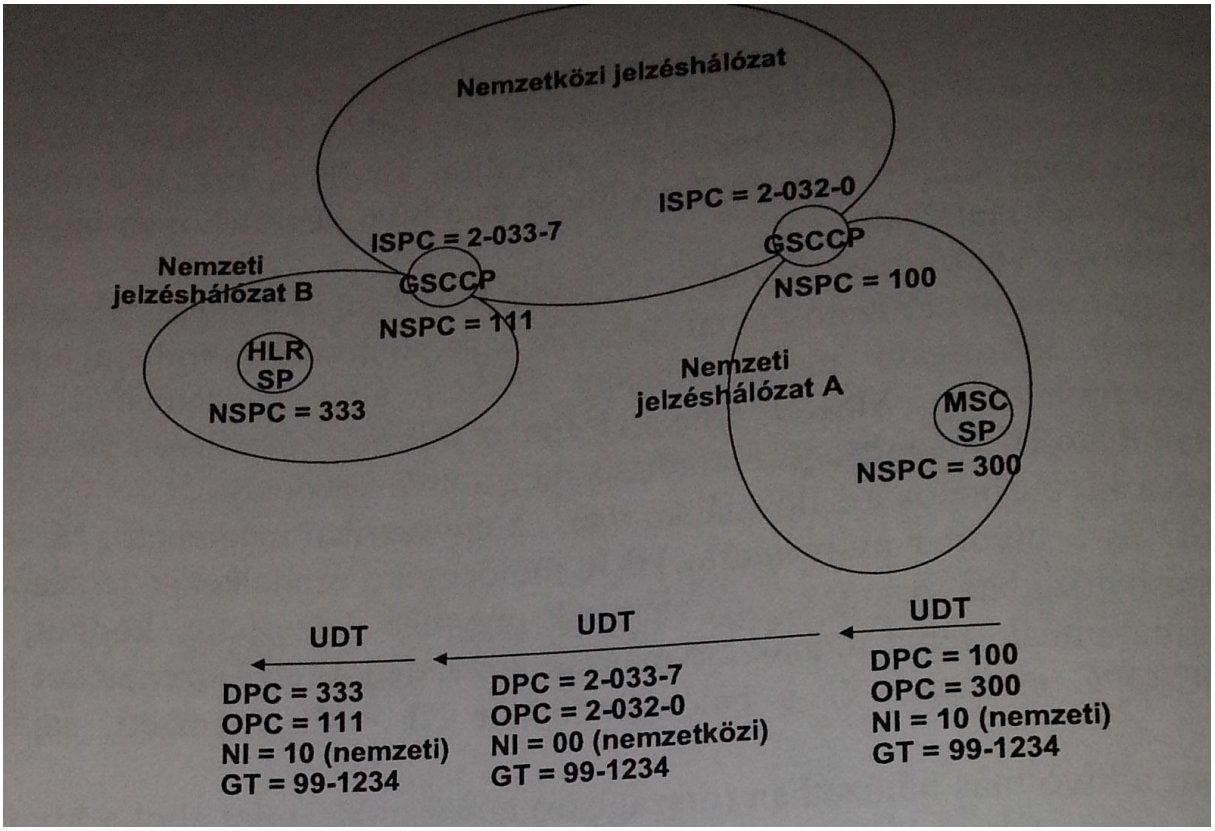
3. Az ábrán bejelölt nemzetközi hívást akarjuk felépíteni. Az ábrán a telefonközpontokat négyzet, a hívás során felhasznált beszédáramköröket vastag vonal, míg az igénybevett jelzéspontokat / jelzéstovábbító pontokat kör, a használt jelzésszakaszokat pedig szaggatott vonal jelzi.
a.) Rendeljen jelzéspont kódokat az egyes jelzéspontokhoz az egyes szakaszokon használt formátumoknak megfelelően! (3p)
b.) Jelölje be, hogy milyen jelzéskapcsolatok épülnek fel! Mi a jelzésüzenetek legfontosabb paramétereinek (DPC,OPC,SIO[NI+SI]) az értéke az egyes szakaszokon? (6p)



Ismertesse két digitális ISDN készülék közötti hívás felépülését, ha a két készülék különböző központban van!

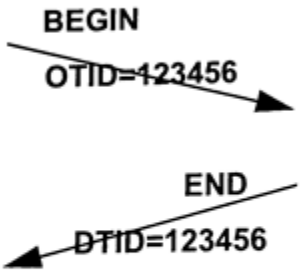


Mutassa be egy példán keresztül, hogy mikor és hogyan használjuk az SCCP golbális címfordító képességét!

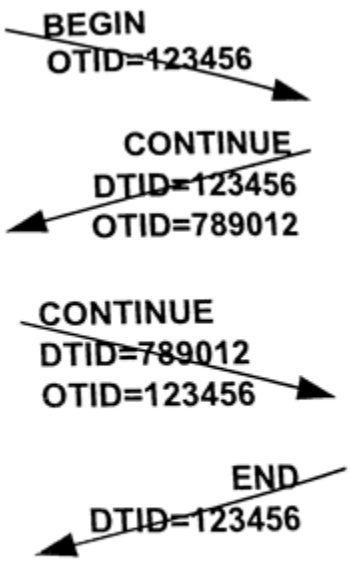


Adjon példát összetett TCAP tranzakcióra. Jelezze a példán használt üzenet, és komponens típusokat, tranzakció és művelet azonosítókat! Hogyan tudunk hibát jelezni TCAP-ben?

/DLR szerepe volt a kapcsolatok azonosí...



77. ÁBRA EGYSZERŰ, EGY BEGIN/END PÁRBÓL ÁLLÓ TRANZAKCIÓ



78. ÁBRA ÖSSZETETT TRANZAKCIÓ

Hogyan zajlik egy tesztlaborban egy protokoll megvalósítás tesztelése (hogyan választjuk ki a futtatandó teszteket, milyen dokumentumok használatosak

Wikiről:
Input: PICS (Protocol Implementation Conformance Statement - protokoll megvalósítás alkalmassági nyilatkozat) és PIXIT (Protocol Implementation eXtra Information for Testing - protokoll megvalósítás vizsgálatára vonatkozó extra információ). Ezek alapján meghatározzák, hogy:

- Mely teszteseteket
- Milyen paraméterekkel
- Milyen elrendezésben

kell tesztelni. Az Abstract Test Suite-ból (absztrakt tesztkészlet) Executable TS-ot (végrehajtható tesztkészlet) készítenek. A futtatás során minden lényeges esemény naplózásra kerül.
Output:
PCTR (Protocol Conformance Test Report - protokoll alkalmassági teszt jelentés):

- Milyen megvalósításon
- Mely vizsgálatokat
- Milyen opciók mellett
- Milyen elrendezésben
- Milyen eredménnyel

hajtottak végre

SCTR (System Conformance Test Report - rendszer alkalmassági teszt jelentés): A teljes vizsgált rendszert jellemzi

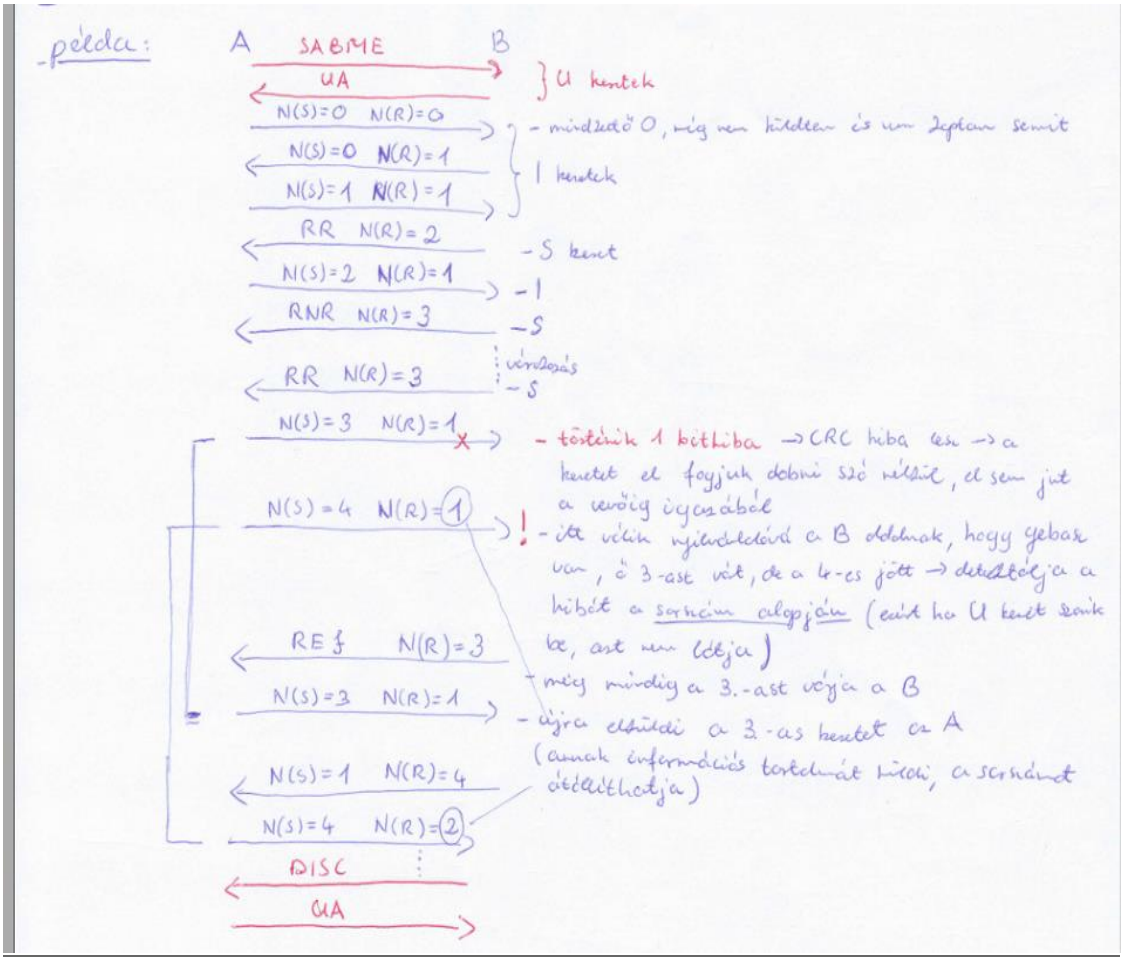
Illetve tesztelés elmélet:
Meg kell tervezni, hogy mit akarunk tesztelni, mivel minden eset tesztelése nem lehetséges a megközelítőleg végtelen számú teszteset miatt. A tesztek lehetséges kimenetei

- Pass: ha minden megfelelő
- Fail: ha nem megfelelő értéket küld
- Inconclusive: ha nem az jött, amit vártunk vagy egyáltalán nem jött semmi, pl. az eszköz jól működik de hálózati hiba miatt nem érkezett meg az üzenet

1. White-box testing: Főleg szoftvertesztelési módszer, amelynél a rendszer belső működését akarjuk tesztelni. Az adott tesztelendő komponenst egy "üvegdobozként" képzeljük el, tehát belelátunk a belső felépítésébe, így tudunk olyan teszteket írni, amely ezt figyelembe veszi. Ilyen tesztelési módszer a bemenetek tesztelése, helyes illetve helytelen valamint szélsőséges esetekben előforduló bemenetekkel futtatjuk, és megnézzük, hogy mit reagál, megfelelőek-e a bemenetek vizsgálata. Továbbá elágazások tesztelése, valóban minden IF ág le tud futni, és nincs benne olyan, amely olyan feltételt vár el, amely sohasem teljesülhet, így gyakorlatilag holtágak maradnak a kódban, amelyek sosem futhatnak le. Valamint a ciklusok tesztelése, nem kerül-e egyik sem végtelen ciklusba, mindegyik lefut-e. Ezzel a 3 módszerrel a kódok 90%-nak lefedettsége elérhető, amely egy jó arány.
    a. Provokatív tesztelés: Azt ellenőrizzük, hogy a hibákat hogyan kezeli, pl. rossz input, rossz sorrend, szintaktikailag helytelen, rosszkor küldünk üzenetet stb..
    b. Function test: egy funkció, modul, szolgáltatás tesztelése eszközön belül
    c. System test: Több funkció együttes tesztelése, hogyan működnek együtt (gyakorlatilag együttműködési teszt más néven)
    d. Regressziós teszt: Akkor kell lefuttatni, ha a belső megvalósítást megváltoztatjuk, ellenőrizni kell, hogy a módosítással nem rontottunk el valamit, így a korábban már helyesen lefutott teszteket a módosítás után ismét lefuttatjuk, hogy továbbra is megfelelően működik-e minden.
2. Black-box testing: Nem ismerjük az adott tesztelendő komponens belső felépítését, számunkra az egy "átláthatatlan" fekete doboz, csak az interfészt (kommunikációs pontjait) "látjuk" és a kimeneten megjelenő eredményt. A bement a Point of Control (PC) és a kimenet a Point of Observation (PO). A szoftverkomponenseken túl, a hardver eszközök tesztelése ilyen, mivel a hardver belső felépítése minket nem izgat, csak azt tudjuk, hogy milyen bemenetet vár el, és, hogy arra milyen kimenetet szeretnénk kapni.
3. Konformancia teszt: Ha egy szoftver már technikailag helyesen működik, akkor utána azt kell ellenőrizni, hogy valóban azt csinálja-e, amit a specifikáció előír (gyakorlatilag verifikáció utáni validáció, tehát nem elég, hogy jól működik, valóban azt is csinálja amire eredetileg készíteni akarták). A konformancia teszt nem állítja, hogy a protokoll megvalósítás 100%-osan konform a specifikációval, a gyakorlatban ez egyfajta elfogadható kompromisszum a tesztesetek nagyon nagy száma miatt. *A konformancia teszt egyfajta black-box teszt, így nem biztos, hogy ez külön típusnak számít.*

   a. Basic Interconnection Test: a minimális követelményeket ellenőrzi, megnézi, hogy a megvalósítás egyáltalán alkalmas-e más megvalósított objektumokkal kapcsolatot felépíteni, kommunikálni. Érdemes-e egyáltalán további teszteket végezni → első szűrő.
   b. Capability Test: ellenőrzi a megvalósított objektumban meglévő statikus tulajdonságokat, képességeket, pl. szabvány szerint előírt kötelező tulajdonságok megvannak-e.
   c. Behavior Test:
4. Együttműködési teszt: A konformancia tesztelésen sikeresen átesett komponensek képesek-e egymással együttműködni.
5. Teljesítőképességi vizsgálat: Mekkora forgalmat generálnak, mekkora forgalmat képesek kezelni, forgalmi csúcsokat hogyan vezetik le, egy adott forgalom mellett mekkora késleltetéssel működnek. Ennek teszteléséhez általában a szimuláció is sokat segít.
   a. Load/Performance test: eltérő terhelések során hogyan viselkedik, mit bír el még az eszköz/rendszer

Mutassa be egy példán keresztül a LAPD hibajavítási rendszerét



ASN1 feladat (zh felkészüléses doksiban van több is)

ZH UTÁNI ANYAGRÉSZES FELADATOK

GSM legfontosabb azonosítói

(Tankönyv 120. oldal)

- **IMSI(International Mobile Subscriber Identity):** a GSM hálózatban használt legfontosabb azonosító a SIM kártyát és azon keresztül az előfizetőt azonosítja. Ez 3 részből tevődik össze: **IMSI = MCC + MNC + MSIN** ahol
  - **MCC(Mobile Country Code):** az ún. mobil országkód, pl. Magyarországé 216
  - **MNC(Mobile Network Code):** az ún. mobil hálózati kód pl. 20,30,70
    így az MCC+MNC együtt azonosítja azt a szolgáltatót, amely a SIM kártyát kiállította, tehát akinél előfizettünk.
  - **MSIN(Mobile Subscriber Identification Number):** max. 10 jegyes mobil előfizető azonosító szám, amely az adott hálózaton belül egyértelműen azonosítja az előfizetőt.

Az IMSI egyértelműen szolgáltatóhoz köthető, tehát ha szolgáltatót váltunk és megtartjuk a hívószámunkat, akkor ezzel SIM kártyát és IMSI-t is cserélni kell. A GSM rendszerekben található adatbázisok (HLR, VLR, AUC..) mindig az IMSI alapján azonosítják a felhasználót, nem telefonszám alapján.

- **MSISDN(Mobile Station ISDN Number):** ez az a telefonszám, amelyen a készülék hívható. Ez a szám 3 részből áll az ITU-T E.164 ajánlás alapján:
  MSISDN = CC + NDC + SN ahol
  - **CC(Country Code):** országkód pl. magyarország 36
  - **NDC(National Destination Code):** szolgáltatás- és hálózatkijelölő szám, amelyet a köznyelvben körzetszámnak neveznek pl. 20,30,70
  - **SN(Subscriber Number):** előfizetői szám, amely jelenleg 7 számjegyű
- **IMEI(International Mobile Equipment Identity):** a mobil készülék azonosítására szolgáló 15 jegyű azonosító. Ez is 3 részből áll: **IMEI = TAC + FAC + SN** ahol
  - **TAC(Type Approval Code):** a készülék típusazonosító kód
  - **FAC(Final Assembly Code):** a gyártó azonosító kód

  E kettő együtt (TAC+FAC) együttesen, egyértelműen azonosít egy készüléktípust, míg a harmadik rész egy hatjegyű gyári szám **SN(Serial Number)**. Ez alapján lehet hálózati bejelentkezéskor ellenőrizni, hogy az adott készülék lopott-e.
- **MSRN(Mobile Station Roaming Number):** a földrajzi hálózatokkal ellentétben a mobil hálózatokban nem lehet telefonszám alapján megmondani, hogy egy készülék hol található, így emiatt bevezették az ún. barangoló számot. Ez alapján lehet hívást irányítani. Ha egy készülék egy adott MSC-hez bejelentkezik, akkor az MSC címe bejegyzésre kerül a HLR-be. Hívásvégződtetéskor a HLR attól az MSC-től ahova az előfizető bejelentkezett, kérni fog egy ideiglenes telefonszámot amelyen az előfizető az adott hívás alkalmával elérhető lesz, ez lesz az MSRN. (ez megy az ISUP IAM hívásfelépítési üzenetében).
- **TMSI(Temporary Mobile Subscriber Identity):** az első bejelentkezéskor a központ hozzárendel a készülékhez egy véletlenszám jellegű ideiglenes azonosítót (TMSI), amely alapján nem visszafejthető, hogy ki kommunikál. Erre a felhasználók kilétének elrejtésére és a kommunikáció titkosítása miatt van szükség (ne lehessen lehallgatni, beazonosítani ki kommunikál). Ezt a TMSI-t a SIM kártyán tárolják. Következő

bejelentkezéskor már ezt használja a készülék. Központ váltáskor az új központ egy újat rendel a készülékhez. Sokszor a gyakorlatban, akár minden beszélgetéskor újat generálnak a nagyobb biztonság érdekében.

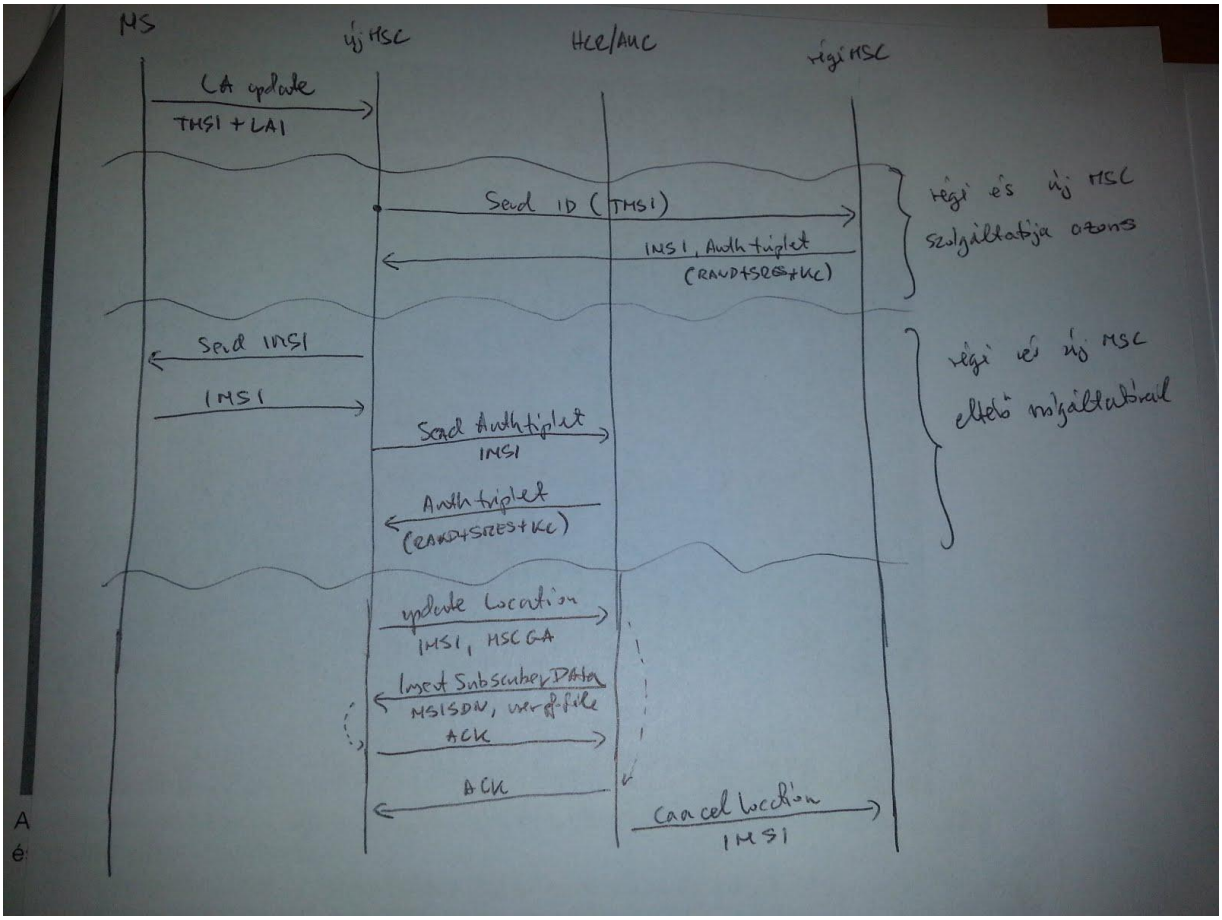
- **LAI(Location Area Identifier):** ún. körzet azonosító, amely szintén 3 részből áll **LAI = MCC + MNC + LAC** ahol az első két tag feljebb már ismertetett, a **LAC(Location Area Code)** pedig a hálózaton belül azonosít egy körzetet és azzal együtt a körzetet vezérlő központot.
- **GCI(Global Cell Identity):** a körzeten belüli cella azonosítására a globális cellaazonosító, amely a körzet azonosító mellett egy cella azonosítót(CI) is tartalmaz. **GCI = LAI + CI**

GSM Location Update NSS-ben (tankönyv 137. oldal)

(wikiről: "A location update-nél mindkét verzió mindig kell, és a TCAP end hiánya miatt, ami mondjuk szerintem tökre nem a lényeghez tartozik szintén bőven lehet pontokat veszteni... Ja és az üzenetekhez kell, hogy mik a paraméterek, ez szintén pontlevonás. ")

Mivel effektíve a location update a kérdés, ez a cella váltásos rész nem biztos, hogy kell

- *A készülék folyamatosan méri az adott cella jelerősségét amelyben tartózkodik, ha a jelszint egy megadott érték alá csökken és egy szomszéd celláé pedig egy érték fölé nő, akkor cellát vált. Erre azért van szükség, hogy ha cella szélén van, akkor kisebb változás miatt ne ugráljon ide-oda a cellákban a mobil.*
  Cellaváltáskor:
  - *Az adott cella ugyanahhoz a LA-hoz tartozik, nem kell tenni semmit*
  - *Az adott cella egy másik LA-hoz tartozik*
  - *Az adott cella egy másik LA-hoz tartozik, amely egy másik MSC-hez tartozik de ugyanahhoz a szolgáltatóhoz*
  - *Az adott cella egy másik LA-hoz tartozik, amely egy másik MSC-hez tartozik és szolgáltató váltás is történt*
- Ekkor ellenőrzi, hogy az adott cella még ugyanahhoz a Location Area-hoz tartozik, vagy LA-t is váltott-e.
  - Ha nem váltott LA-t, akkor ugyanahhoz az MSC-hez tartozik, akkor az MSC a saját VLR-jében átírja a LA-t (IntraLA váltás)
  - Ha váltott LA-t, akkor (InterMSC váltás) az új LA más MSC-hez tartozik mint az előző, az MSC-nek meg kell tudnia a készüléktől az IMSI-jét, kell AuthTriplet, kell készülék UserProfile (a mobil TMSI-vel jelentkezett be). HLR-hez el kell küldeni az új MSC azonosítót, értesíteni a régi MSC-t, hogy törölheti a VLR-jéből a készüléket



Az ábra első részében a régi és új központ azonos szolgáltatónál. Második esetben pedig a régi és új MSC eltérő szolgáltatónál.

1. A készülék elküldi a TMSI-jét és a régi LAI-t, itt a LAI-ból kiderül, hogy ugyanahhoz az MSC-hez tartozik vagy sem, illetve volt-e szolgáltató váltás vagy sem (mivel a LAI = MCC + MNC + LAC)
2. a TMSI alapján elkéri a régitől az IMSI-t és az autentikációs tripletet (RAND+SRES+Kc). Az IMSI-ben benne van az ország és szolgáltatókód (IMSI = MCC + MNC + MSIN),
3. Mivel az új MSC nem ismeri a régi MSC-t a szolgáltató váltás miatt, ezért elkéri a készüléktől az IMSI-t, amely alapján beazonosítja a honos szolgáltatónál a HLR-t.
4. Az új MSC a HLR-től elkéri az autentikációs tripleteket, majd lezajlik az autentikációs folyamat (másik feladatnál ott van részletesen az ábra)
5. Az új MSC a HLR-nek elküldi a location update-et a készülék IMSI-jével és a saját globális azonosítójával, amelyet a HLR nyugtáz és az MSC is nyugtáz
6. Végül szól a HLR, hogy törölheti a régi MSC a VLR-ből a készüléket

Gyakorlatban, hogy spóroljanak a rádiós csatorna terhelésével, az eszközök LA update során mindig IMSI-t küld amúgy meg TMSI-t, így egy plusz üzenetküldés megspórolható. Ha nincs pozíció váltás, akkor is van periodikus LA update, hogy a HLR tudja frissíteni az adatait, abban az esetben, ha valamikor HLR leállás lett volna, de ezeket a HLR-ek eldobják általában, kivéve ha tényleg volt leállás.

GPRS-ben miért kell csökkenteni a paginget, és hogyan oldják meg?

http://books.google.hu/books?id=uxynrOg6BFcC&pg=PA92&lpg=PA92&dq=gprs+routing+area&source=bl&ots=jViWgQ7Yeb&sig=W5xKfmYd4alq7pvkKLyZATtdcdM&hl=hu&sa=X&ei=EiiCU6__GsLD7AaMlYGwBg&ved=0CFIQ6AEwAw#v=onepage&q=gprs%20routing%20area&f=false

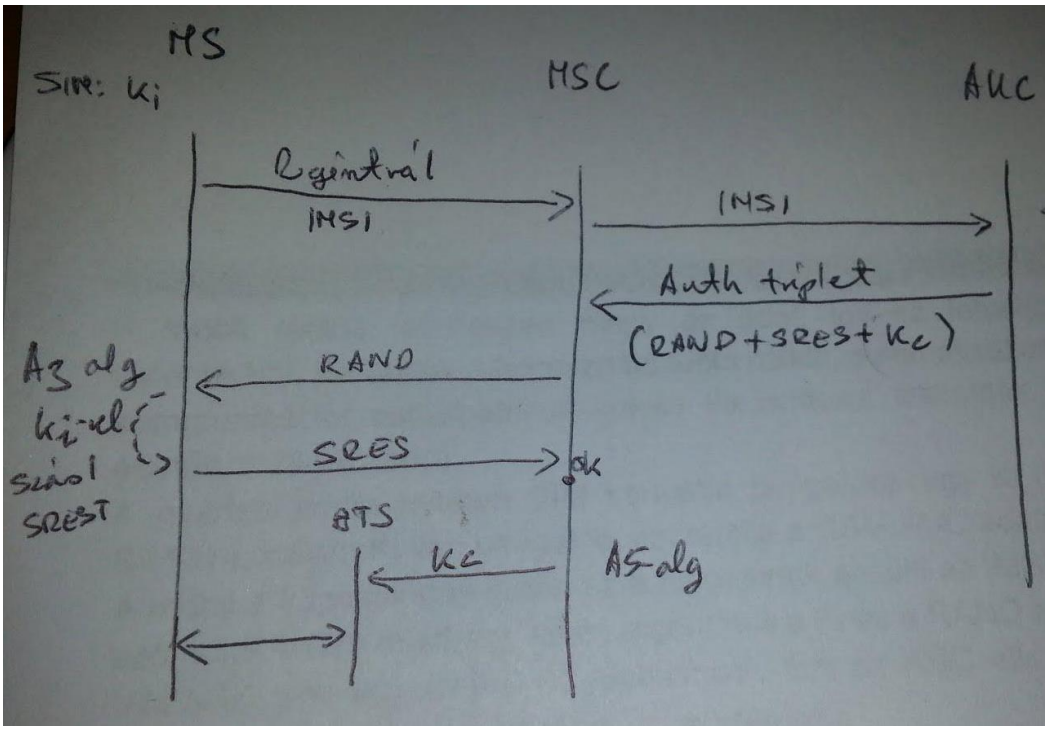
Az adatforgalom jellemzése teljesen eltérő mint a beszédé. A beszéd rövid ideig tart, illetve 1 beszéd alatt kevesebb cellaváltás történik, de van rá handover, illetve folyamatosan tart, valamint nem gond, ha néhány csomag elveszik, más QoS van, pl ne legyen késleltetés ingadozás.
Az adatforgalom BURST jellegű és nem folyamatos, (nem veszhetnek el csomagok) így nagy valószínűséggel mozog el a cellából a következő burst-ig, hosszú szünetek vannak a burst-ök között. (pláne kis UMTS/LTE cellák esetén)
Nem célszerű végig követni a felhasználót, ahogy beszédnél sem volt, csak a burst elején keressük meg (sok burst, sok paging). A sok paging megspórolása miatt bevezették a **ROUTING AREA (RA)** azonosítót, amely kisebb területet fed le mint a LA. A beszédet 1 LA-ban építik fel, míg az adatkapcsolatot 1 RA-ban. tehát **RA < LA** ahol **RAI = LAI + RAC**
A mobilnak az RA váltást kell jeleznie, ezért elég csak az SGSN-hez bejelentkezni, mert pontosabban ismeri a helyzetét mint az MSC. Az SGSN-hez RA update során felküldi a **RAI+PTMSI**-t ahol PTMSI a Packet-TMSI.
Az RA update teljesen analóg a LA update-tel.
Beszédet tehát LA-ban építünk fel, adatkapcsolatot pedig RA-ban.
Definiálták a **NullRA-t,** amely olyan körzet, ahol nincs GPRS szolgáltatás

Továbbá a hálózat nyilvántartja, hogy a mobil milyen állapotban van:

- **Idle:** mobil nem kapcsolódik az adathálózathoz, így LA szinten tartjuk nyilván
- **Ready:** kereteket küld, cella szinten ismerjük a helyzetét
- **Standby:** ha befejeződik az adatforgalom, akkor indul egy timer, ha az lejár, menjen át standby-ba és cella helyett csak RA szinten tartjuk nyilván a pozícióját

Mobil állomás autentikálása, miért van rá szükség és mikor, hogyan történik?

A mobil rádiós interfészén megy az adat, így ez lehallgatható lenne, ezért szükséges autentikálni, de ennek a felhasználó tudta nélkül, automatizáltan kell történnie. Továbbá az egész kommunikációt aszimmetrikus kulcsú titkosítással titkosítják is. (GSM esetén 2^128, UMTS esetén 2^256 db kód)
A gyártás során minden SIM kártyába beégetnek egy K_i (Individual SubscriberKey) -t, adott SIM-hez milyen Ki van beégetve, ezt tárolja a GSM AUC adatbázisában.



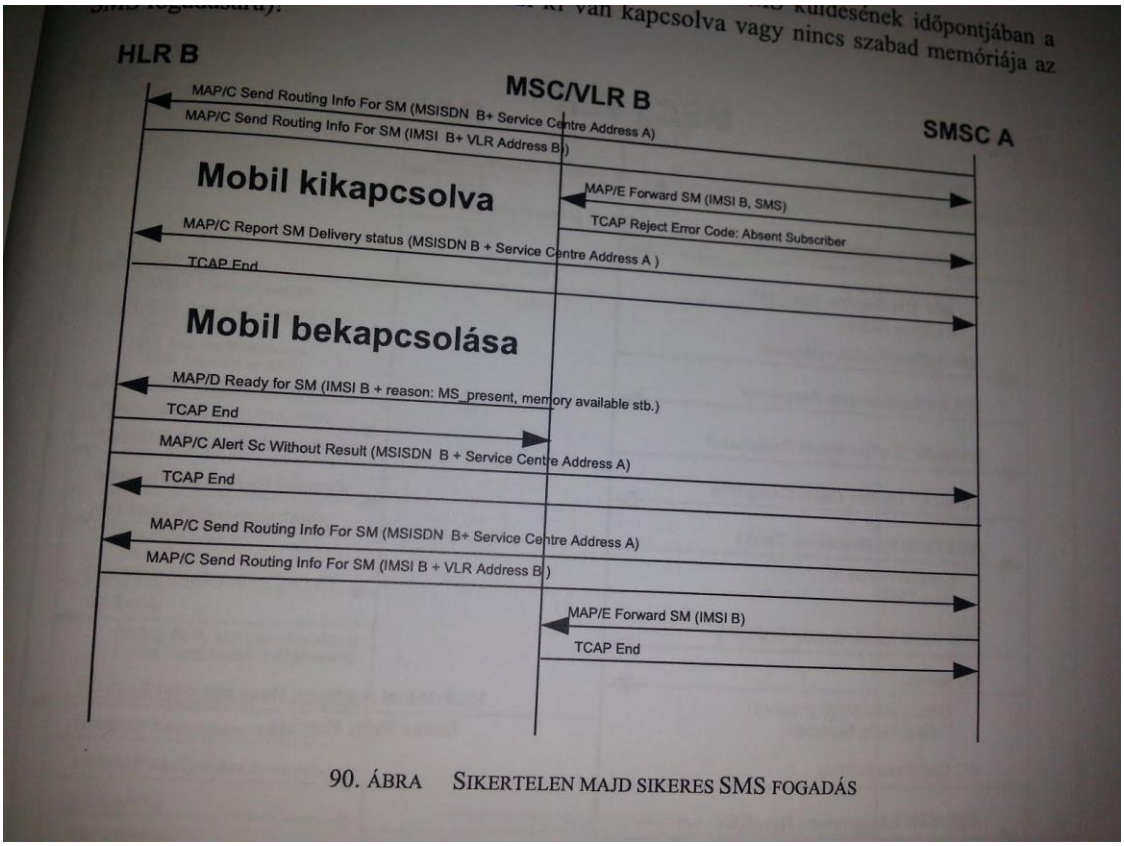
1. Először a mobil “regisztrál” LA update-tel az MSC-nél, az MSC elküldi az IMSI-t az AUC-nak és az alapján kér autentikációs adatokat, vagyis Autentikációs Tripletet (azért triplet, mert három paramétert kap vissza)
2. Az AUC az IMSI alapján elvégzi a számításokat és az autentikációs tripletet visszaküldi (RAND+SREC+Kc).

Minden SIM kártyán található egy egyedi Ki kód (GSM esetén 2^128, UMTS esetén 2^256 kódú), amelyet a szolgáltató az AUC-ban rögzít az IMSI-hez. Az autentikációs triplet előállítása során az AUC generál egy véletlenszámot, amely a RAND lesz, ekkor a RAND és az IMSI-hez tárolt Ki kód alapján kiszámít egy SRES értéket. Az MSC ezt megkapja, a RAND-ot elküldi a mobil készüléknek, akinek ugyanez a Ki értéke van a SIM kártyán, tehát ugyanazt az SRES értéket kell generálnia. Miután a mobil ezt kiszámolta, felküldi az MSC-nek, aki bit szintű egyezést vizsgál.

3. Az MSC elküldi a mobilnak a RAND-ot, hogy számítsa ki a saját SRES értékét
4. A mobil kiszámolja a választ (SRES), amit a központ ellenőriz (bitenkénti összehasonlítás) és ha megfelelő, akkor engedélyezi (A3-alg. végzi a számítást)
5. Az MSC a BTS-nek elküldi a harmadik paramétert, a Kc kódot, amely a kommunikáció titkosításához szükséges. (A Kc kód alapján az A5-algoritmus végzi a titkosítást)
6. A titkosított beszédátvitel csak a mobil és a BTS között van, a BTS végzi el

Ismertesse egy SMS útját a feladó központjától a címzett központjáig

SMS elmélet: jelzés szolgáltatás amely gyakorlatilag egy egyszerű jelzés üzenet, egyáltalán nem garantált a kézbesítése, nincs garancia a megérkezésére. Az SMS Datagram szolgáltatás, tehát nincs kapcsolat kiépítve a küldő és a címzett között. Ezért is van itt egy mobil kikapcsolt és bekapcsolt rész, mert a hálózat többször is újra próbálja periodikusan küldeni az üzenetet, ha a készülék ki volt kapcsolva, illetve ha bejelentkezik.

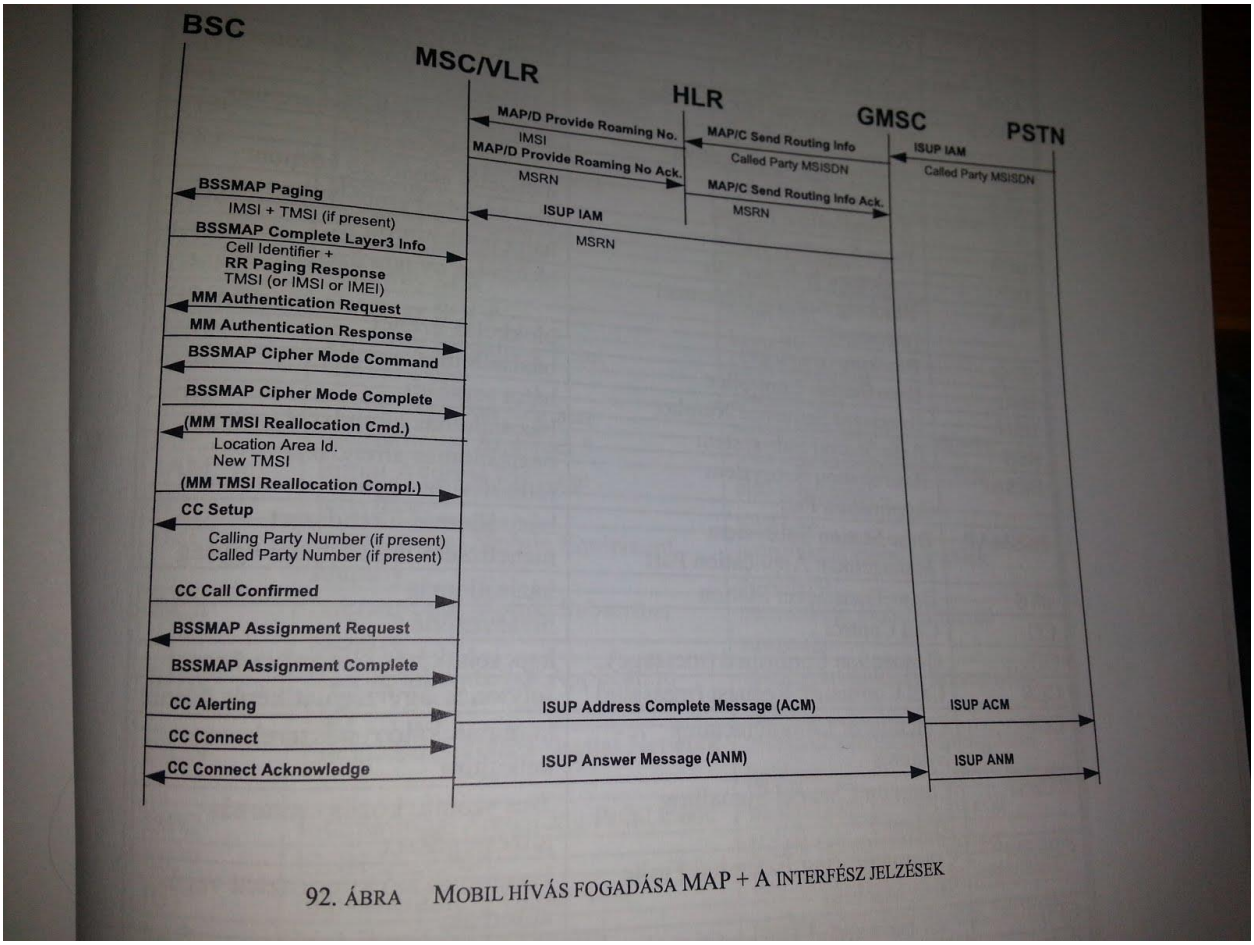


90. ÁBRA SIKERTELEN MAJD SIKERES SMS FOGADÁS

Az ábrán a nyilakon lévő MAP/C rész nem kell, csak utána az üzenet neve és a paraméterek.
A: küldő adatai
B: címzett adatai

1. Az SMS feladójának MSC-je elküldi a kapott SMS-t a feladó SMS központjához, az SMSC -hez
2. A feladó SMSC-je a címzett HLR-jéhez fordul routing információért, amelyre válaszban a készülék IMSI-jét és az MSC MSISDN GT-jét kapja vissza
3. A feladó SMSC-je megpróbálja a címzett MSC-jének továbbítani az SMS-t, de az jelzi, hogy a mobil nem tudja fogadni, mert vagy ki van kapcsolva, vagy megtelt a memóriája
4. A feladó SMSC-je megkéri a címzett HLR-jét, hogy ha a mobil elérhetővé válik, akkor jelezze, ekkor majd újra megpróbálja elküldeni
5. A címzett MSC-je egy ReadyForSM üzenettel szól a HLR-nek, hogy a mobil elérhető
6. A HLR egy AlertInfoForSM üzenettel jelzi a feladó SMSC-jének, hogy próbálja meg ismét a küldést mert a mobil elérhető. Innetől ismét a fenti fázis fut le.

Ismertesse egy mobil készüléken végződő hívás (MT call) felépülését az A interfészen!



92. ÁBRA MOBIL HÍVÁS FOGADÁSA MAP + A INTERFÉSZ JELZÉSEK

A lenti szöveg nem egészen nyerő, mert az A interface az a BSC és az MSC/VLR közti szakasz. Tehát pont nem az A if-ről ír.

A mobil telefonszáma (MSISDN) alapján csak a szolgáltató hálózatának Gateway-éig lehet eljutni, hálózaton belül már nem a telefonszám alapján történik az útirányítás, hanem az ún. MSRN alapján.

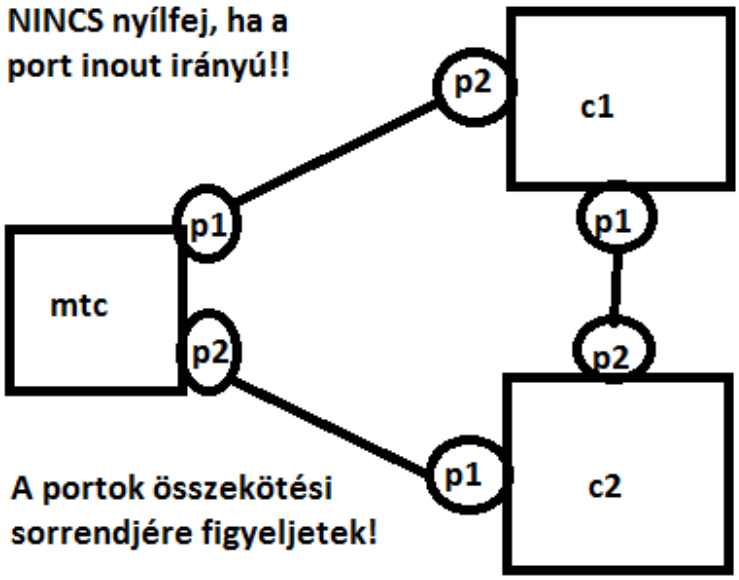
1. Beérkezik az MSISDN alapján a hívás a Gateway-hez, ekkor a GW a HLR-hez fordul, hogy az MSISDN alapján kérjen attól az MSC-től egy MSRN azonosítót, akinél jelenleg a mobil található
2. A HLR tárolja, hogy melyik MSISDN-hez melyik IMSI tartozik, és, hogy az adott IMSI-hez tartozó készülék melyik MSC-nél van (ez a VLR-ben van), így az adott MSC-hez fordul az IMSI-vel, hogy az IMSI-hez tartozó MSRN-t küldjön
3. Az MSC elküldi a HLR-nek az MSRN-t, aki azt továbbítja a GMSC-nek, aki így már tudja irányítani a hivást, és felépíti a kommunikációs utat az MSC-vel
4. MSC és a mobil között lezajlik a paging, megkell találni a készüléket és szólni neki, hogy bejövő hívása van. Miután a készülék vissza jelzett, lezajlik az autentikációs folyamat és utána a kommunikáció titkosítása a mobil és a BTS között
5. Ezután kezdődik a hívás felépítése a SETUP üzenettel (ISUP üzenetek, ahogy korábban már tanultuk vonalas esetben a hívás felépítést)

TTCNv3 feladat

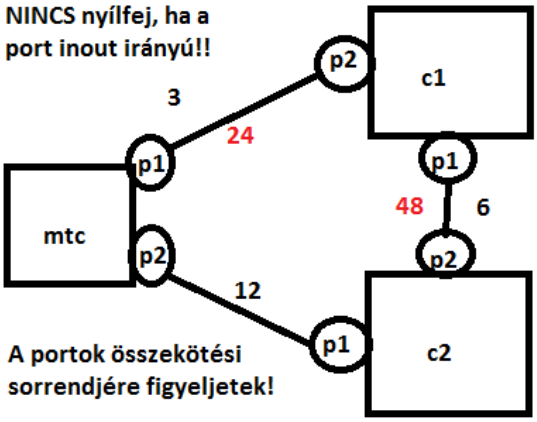
TTCNv3 kód

```
module vizsgal {

type port PortType message { Inout integer, charstring;)

type component CompType {
   port PortType p1;
   port PortType p2;
   timer T := 10.0;
   var integer temp; }

template integer GoodNumber := ((1..20), 24, 50);

function f() runs on CompType {
   T.start; alt {
      [] p2. receive (GoodNumber) -> value temp ( p1. send (temp *2); setverdict
(pass) ; repeat }
      [] p2.receive { setverdict(fail); mtc.stop }
      [] T.timeout { setverdict(inconc); }
   }
}

testcase tc() runs on CompType {
   var CompType cl := CompType.create;
   var CompType c2 := CompType.create;

   connect(mtc:p1, cl:p2);
   connect(cl:p1, c2:p2);
   connect(c2:p1, mtc:p2);

   cl.start(f());
   c2.start(f());
   p1.send(3);
   T.start;
   alt {
      [] p2. receive (GoodNumber) -> value temp { p1. send (temp *2); setverdict
(pass) ; repeat }
      [] p2.receive { setverdict(fail); self.stop }
      [] T.timeout { setverdict(inconc); }
   }
   all component.done;
}

control {
   execute(tc());
}
}
```

a.) Rajzolja fel a teszt komponenseket, portokat és azok kapcsolatait.



b.) Mutassa meg a teszt folyamatát! (Rajzoljon!)



Sorrend: 3, 6, 12, 24, 48

48 az nincs benne a Good Number korlátozott típusban!
Ezért p2.receive { setverdict(fail); mtc.stop } fut le.
mtc.stop = Main Test Case lelövése

(De közben volt: none, pass, ..., pass majd fail)

c.) Mi a tesztesemény kimenete?

FAIL, mert a 48 nem egy elfogadott érték (nincs a Goodnumber elemei között)

Egy másik feladat, ugyanezekkel a kérdésekkel!

A feladat kódja:

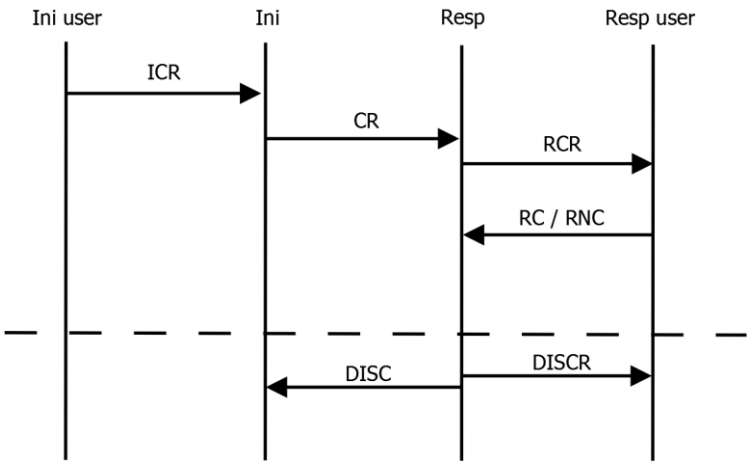
```
type port PortType message {
    inout integer, charstring;
} with { extension "internal" }

type component CompType {
    port PortType P;
    timer T := 10.0;
}
template integer GoodNumber := ((5..11), 13);

function f() runs on CompType
{
    T.start;
    alt {
    [] P.receive(GoodNumber) { P.send("BINGO"); repeat; }
    [] P.receive("HELLO") { P.send(5); repeat; }
    [] T.timeout { setverdict(inconc); }
    }
}
altstep MyAltstep(PortType X) runs on CompType
{
    var integer temp;
    [] X.receive(integer:?) -> value temp { X.send(temp * 2); repeat; }
    [] X.receive { setverdict(fail); self.stop; }
}
testcase tc() runs on CompType
{
    var default d := activate(MyAltstep(P));
    var CompType c := CompType.create;
    connect(mtc:P, c:P);
    c.start(f());
    P.send("HELLO");
    P.receive("BINGO");
    setverdict(pass);
    c.done;
}
control {
    execute(tc());
}
}
```

Megoldás:

SDL diagram rajzolás

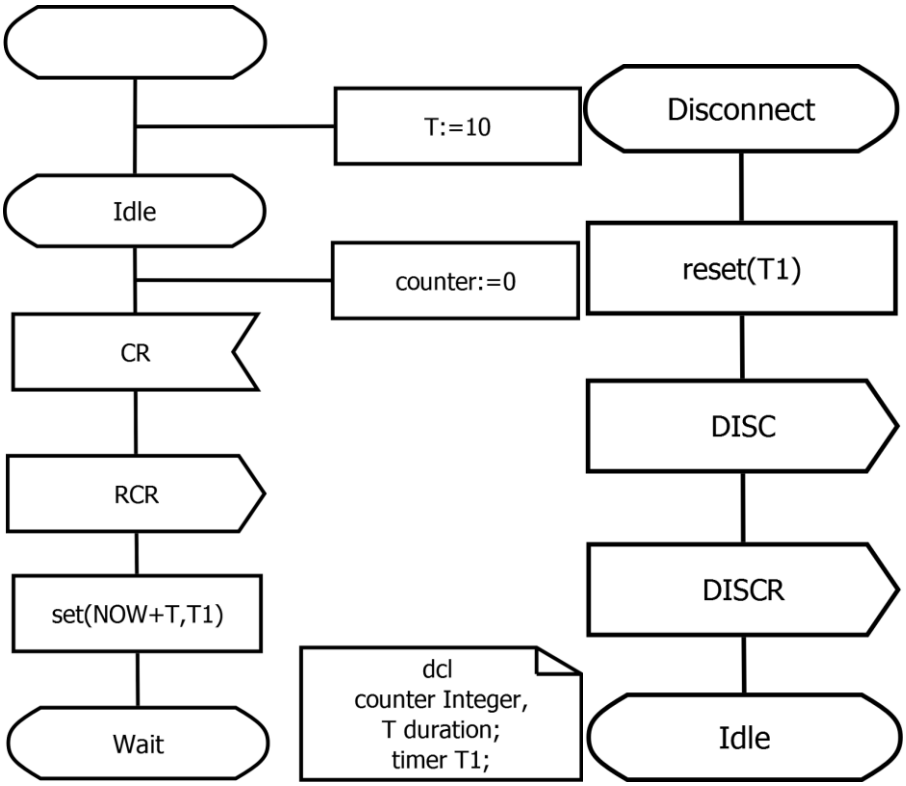
(wikiről: Az SDL diagramnál minden állapot kezdete után az első lépés egy bejövő üzenet kell, hogy legyen, ha ezt nem követed, de egyébként jó a logikád és amúgy működne is csak egyszerűsíteni akartál, akkor "ez nem SDL diagram" --> -7 pont a 10-ből.)

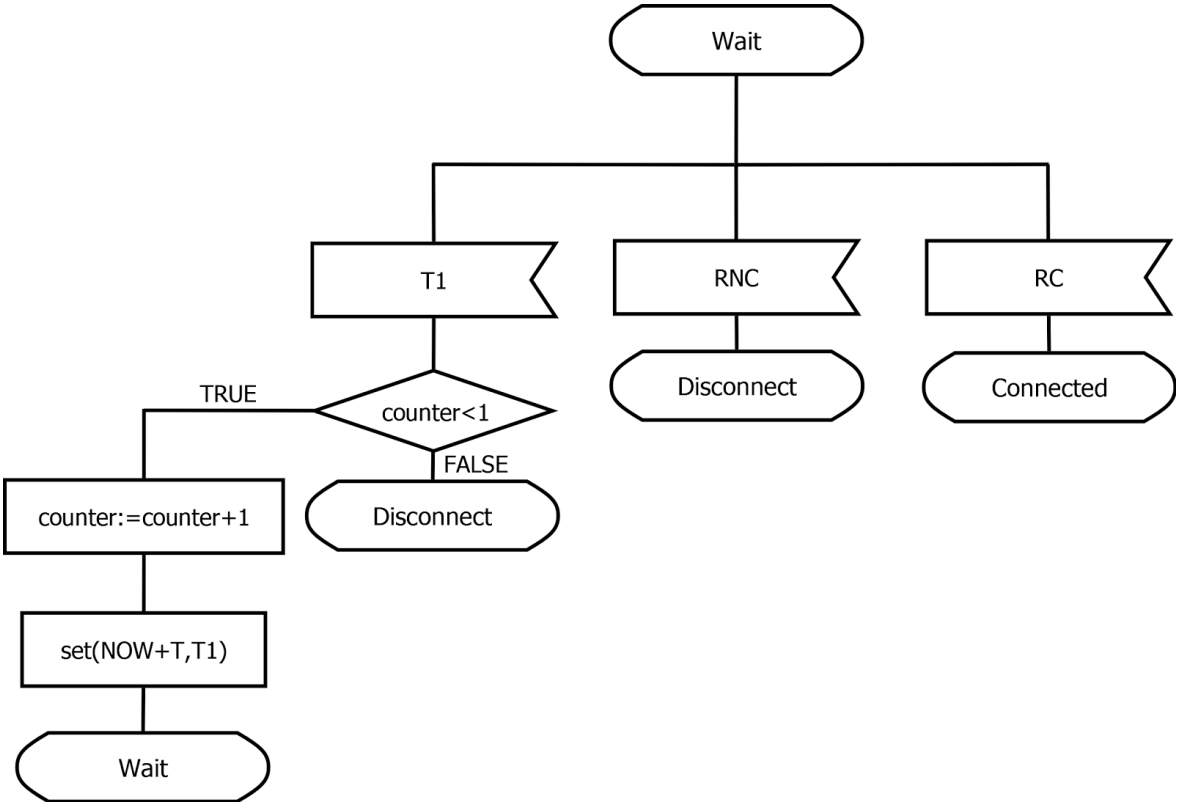
Valósítsa meg az ábrán látható "Resp" funkciót SDL diagrammal. A folyamatábrát egészítse ki szétkapcsolással és ezt is építse bele a diagrammba.



A feladat leírása: A vevő oldalon, egy CR üzenetre várunk, amely hatására kiadjuk az RCR üzenetet. Erre a válasz vagy egy RC vagy egy RNC üzenet lehet. Emellett a CR üzenet érkezésekor beállítunk egy timer-t, amely 10 másodperc múlva jár le. Az első time-out után tovább várunk, a másodiknál bontjuk a kapcsolatot.

Megoldás:




Wait
T1
RNC
RC
Disconnect
Connected
TRUE
counter<1
FALSE
counter:=counter+1
Disconnect
set(NOW+T,T1)
Wait

2014.05.28-ai vizsga

Protokoll technológia vizsga 2014. máj. 28.

NÉV:_____

NEPTUN KÓD:_____

1. Milyen műszaki megoldások használhatók Magyarországon a vezetékes számhordozás megvalósítására? Ismertesse őket röviden! Röviden magyarázza el, miért nem alkalmazhatók a vezetékes hálózati megoldások a mobil hálózatokban! (4p)

2. Ismertesse egy Location Update lefolyását az NSS-ben! (4p)

3. Miért kell és hogyan lehet GPRS hálózatokban a Paging forgalmat csökkenteni? (4p)

4. A következő ASN.1 leírás egy MAP üzenet (általam kissé módosított) definíciójának részlete:

```
send1 OPERATION
        ARGUMENT
                routingInfoForSM-Arg [31] SEQUENCE {
                        msisdn [1] IMPLICIT OCTET STRING (SIZE (1..9)),
                        sm-RP-PRI [3] BOOLEAN,
                        serviceCentreAddress [22] BIT STRING (SIZE (1..20)),
                        teleservice [5] OCTET STRING (SIZE (1)) OPTIONAL}
::= localValue 45
```

A vonalon a következő értékeket szeretnénk kiküldeni:

msisdn: '212121'O
sm-PR-PRI: FALSE
serviceCentreAddress: '0101010101'B

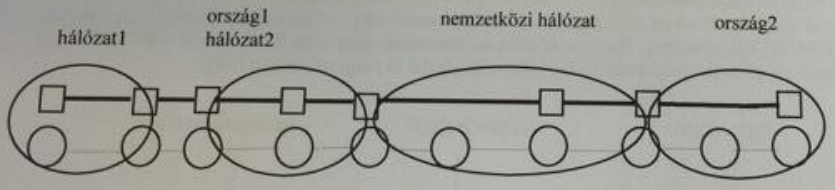
a.) Milyen típusú TCAP keretben/komponensben küldhetjük a fenti üzenetet? (2p)
b.) Készítse el a fenti üzenet MAP részének kódolását! **Mindenhol, ahol lehet, indefinit hossz kódolást használjon!** (10p)
Segítségül:

Class:
- **00 = Universal**
- **01 = Application wide**
- **10 = Context-specific**
- **11 = Private**

Format:
- **0 = Primitive**
- **1 = Structured**

TAG kódok:
- **1 = BOOLEAN**
- **2 = INTEGER**
- **3 = BIT STRING**
- **4 = OCTET STRING**
- **5 = NULL**
- **10 = ENUMERATED**
- **16 = SEQUENCE / SEQUENCE OF**
- **17 = SET / SET OF**

5 Az ábrán bejelölt nemzetközi hívást akarjuk felépíteni. Az ábrán a telefonközpontokat négyzet, a hívás során felhasznált beszédáramköröket vastag vonal, míg az igénybevett jelzéspontokat / jelzéstovábbító pontokat kör, a használt jelzésszakaszokat pedig szaggatott vonal jelzi.
a.) Rendeljen jelzéspont kódokat az egyes jelzéspontokhoz az adott helyen használatos formátumban!
b.) Jelölje be, hogy milyen jelzéskapcsolatok épülnek fel! Mi a jelzésüzenetek legfontosabb paramétereinek (DPC,OPC,SIO[NI+SI]) az értéke az egyes szakaszokon? (6p)



6. Adott a következő TTCN-3 program:

```
module vizsga1 {

type port PortType message {
    inout integer, charstring;}

type component CompType {
    port PortType p1;
    port PortType p2;
    timer T := 10.0;
    var integer temp;}

template integer GoodNumber := ((1..20), 24, 50);

function f() runs on CompType{
    T.start;
    alt {
    [] p1.receive(GoodNumber) -> value temp { p2.send(temp * 2); setverdict(pass); repeat }
    [] p1.receive { setverdict(fail); mtc.stop }
    [] T.timeout { setverdict(inconc); }
    }
}

testcase tc() runs on CompType{
    var CompType c1 := CompType.create;
    var CompType c2 := CompType.create;
    connect(mtc:p1, c1:p2);
    connect(c1:p1, c2:p2);
    connect(c2:p1, mtc:p2);
    c1.start(f());
    c2.start(f());
    p2.send(3);
    T.start;
    alt {
      [] p1.receive(GoodNumber) -> value temp { p2.send(temp * 2); setverdict(pass); repeat }
      [] p1.receive { setverdict(fail); self.stop; }
      [] T.timeout { setverdict(inconc); }
    }
    all component.done;
}

control {
   execute(tc());
}
}
```

Kérdések:

a.) Rajzolja fel a tc teszteset futása során létrejövő teszt konfigurációt (teszt komponenseket, portokat és a köztük lévő kapcsolatokat)! (3p)

b.) Milyen üzenetváltás játszódik le a teszteset futtatása során a teszt komponensek között? Készítsen ábrát! (5p)

c.) Milyen ítéletet hoznak az egyes komponensek és milyen ítélettel zárul a teszteset? (2p)

7. Készítse el a következő protokoll RESPONDER processzének SDL diagramját a kapcsolatfelépítésre vonatkozóan! Ha a kapcsolat kérésre (RCR – Responder Connection Request) T=10 µs időn belül nem érkezett pozitív (RC – Connection Confirm) vagy negatív (RNC) nyugta, akkor a kérést a RESPONDER újraküldi. Ha ez a kísérlet is sikertelen, vagy újra RNC-t vesz, a RESPONDER kezdeményezzen bontást! Tegyen javaslatot a bontási folyamatra! (Rajzolja be az ábrába és magyarázza el!) (10p)