Definícia veľkosti paketov (Kap. 2.4.5.3)

Pri návrhu ATM bolo treba určiť, či sa budú používať pakety pevnej alebo variabilnej dĺžky a taktiež špecifikovať ich veľkosť (to v prípade, ak sa rozhodneme pre pevnú dĺžku), alebo rozsah ich veľkosti (pre prípad výberu variabilnej dĺžky paketov). Ukážeme si, že tento výber ovplyvnilo viacero faktorov.

Variabilná verzus fixná dĺžka paketov

Prvotná definícia ATD (*Asynchronous Time Division*) uvažovala iba o veľmi krátkych paketoch pevnej dĺžky (16 bytov). Naproti tomu FPS (*Fast Packet Switching*) bolo zase navrhnuté pre pakety variabilnej dĺžky. Najhlavnejšie faktory, ktoré ovplyvňujú výhody a nevýhody oboch riešení sú využitie šírky pásma, výkonnosť prepínania (tu treba brať ohľad aj na zložitosť prepínačov) a oneskorovanie.

Využitie šírky pásma

Pretože pakety obsahujú aj hlavičku, efektívnosť prenosu vypočítame ako

$$\eta = \frac{\#\text{ dátových bytov}}{\#\text{ dátových bytov} + \#\text{ bytov hlavičky}}$$

- **Pakety pevnej dĺžky**

Vďaka tomu, že pakety majú pevnú dĺžku, efektívnosť prenosu vypočítame ako

$$\eta_F = \frac{X}{\left\lceil \frac{X}{L} \right\rceil .(L+H)}$$

kde L = Veľkosť dátovej časti paketu v bytoch
H = Veľkosť hlavičky v bytoch
X = Celková veľkosť prenášaných dát v bytoch

Efektívnosť je optimálna pre násobky veľkosti dátovej časti paketu t.j. $\left\lceil \frac{X}{L} \right\rceil .L = X$.

V optimálnom prípade efektívnosť vypočítame $\eta_{FOPT} = \frac{L}{L+H}$.

Efektívnosť teda závisí od veľkosti prenášaných dát – pri veľkom dátovom bloku možno dosiahnuť optimálnu efektívnosť. Naproti tomu pri použití malého bloku dát je efektívnosť dosť nízka. Posúďme teraz efektívnosť prenosu vzhľadom na možné aplikácie:

- *Prenos zvuku* – je to CBR (Continuous Bit Rate) služba. Ak chceme dosiahnuť optimálnu efektívnosť, musíme kompletne vypĺňať pakety, čím do prenosu zvuku zavedieme akési oneskorenie. Čím bude veľkosť paketu väčšia, väčšie bude aj oneskorenie.
- *Prenos videa* – ak použijeme kódovanie s konštantným dátovým tokom, dostávame situáciu ako pri prenose zvuku. Pri kódovaní s variabilným dátovým tokom môže nastať situácia, keď dátový paket nebude vyplnený úplne, čo znižuje efektívnosť prenosu. Avšak pri prenose videa sa jedná o veľké množstvá dát, a teda dospejeme skoro k optimálnej efektívnosti.
- *Dáta* – tu je rozdiel medzi nízko a vysokorýchlostnými aplikáciami. V prvom prípade (napr. vstup z klávesnice) sa dosahuje pomerne malá efektívnosť (10%). Pri vysokých rýchlostiach (napr. FTP) je opäť vysoká efektívnosť, napríklad prenos 1000b súboru sa dosahuje efektívnosť 89%.

Keďže prenos vo vysokorýchlostných sieťach bude tvoriť prenos zvuku, videa a dát, celková efektívnosť bude optimálna aj pri použití paketov pevnej dĺžky.

- **Pakety variabilnej dĺžky**

Pri variabilnej dĺžke paketov musíme do hlavičky zahrnúť aj pár bitov na indikáciu dĺžky prenášanej informácie, prípadne aj kontrolný bit na detekciu chýb. Efektívnosť prenosu potom vypočítame ako $\mu_V = \frac{X}{X + H + h_v}$, kde h_v sú spomínané pridané bity. Pri prenose paketmi variabilnej dĺžky dosahujeme opäť veľmi vysokú efektívnosť (skoro 100%), a to hlavne pre veľké pakety.

- **Záver**

S ohľadom na efektívnosť prenosu môžeme skonštatovať, že pakety variabilnej dĺžky poskytujú lepšiu efektívnosť ako pakety pevnej dĺžky. Avšak ak sa pozrieme na konkrétne využitie širokopásmových sietí, zistíme, že tento prínos nie je až taký výrazný.

Rýchlosť prepínania a jeho zložitosť

Zložitosť prepínania paketov pevnej alebo variabilnej dĺžky závisí na funkciách, ktoré chceme vykonávať, a na ich technologických požiadavkách. Najdôležitejšie faktory, ktoré ovplyvňujú zložitosť prepínačov sú rýchlosť operácií a veľkosť pamäte použitej na frontu v nich.

- **Rýchlosť operácií**

Rýchlosť závisí na vykonávaných funkciách a čase dostupnom na ich vykonanie.

- *Spracovanie hlavičky*

Spracovanie hlavičky je dôležitou časťou ATM prepínačov. Predpokladajme identické funkcie hlavičky pri paketoch ako pevnej, tak i variabilnej dĺžky. V prvom prípade pri paketoch veľkosti 48 + 5 bytov musí prepínač spracovať hlavičku za 2.8 μs, ak chce dosiahnuť rýchlosť 150 Mbits/s.

Pri variabilnej dĺžke paketov musíme zobrať do úvahy najhorší prípad (t.j. 5 + 5 bytov) pri rýchlosti 150 Mbits/s, z čoho dostávame iba 533 ns.

- *Manažment fronty*

V prípade paketov pevnej dĺžky je tu situácia jednoduchšia, pretože bloky v pamäti sú rovnako veľké. Preto stačí použiť jednoduché algoritmy na správu pamäte. Pri variabilnej dĺžke paketov už treba použiť rýchle implementácie algoritmov FindBest, FindFirst a podobne. Tu je tiež náročnejší aj pamäťový manažment.

- **Veľkosť pamäte použitej na frontu**

Pamäťové nároky systému pre pakety s pevnou dĺžkou závisia od zaťaženia siete a prijateľného stratového pomeru. Pre systém s variabilnou dĺžkou paketov je určenie veľkosti pamäte ešte komplikovanejšie a závisí aj od aktuálneho mixu dĺžok paketov. Opäť je najjednoduchšie uvažovať o najhoršom prípade, teda pre najväčšiu dĺžku paketu. Preto pri tomto pohľade je výhodnejšia pevná dĺžka paketu oproti dlhým paketom variabilnej dĺžky. Lepší odhad možno urobiť len veľmi ťažko, pretože nepoznáme aktuálny mix dĺžok paketov.

- **Záver**

Obidva parametre, t.j. rýchlosť operácií aj veľkosť pamäte sa dajú lepšie dosiahnuť pri použití paketov pevnej dĺžky.

Oneskorenie

Je dobré, keď veľkosť ATM paketov nie je príliš veľká, aby sa do systému nevnieslo príliš veľké oneskorovanie, čo je nevhodné zvlášť pre systémy bežiace v reálnom čase (napr. prenos zvuku).

Súhrn o použití pevnej alebo variabilnej dĺžky

Pre širokopásmové siete, ktoré sú určené hlavne na prenos zvuku, videa a dát, je prínos z použitia paketov variabilnej dĺžky zanedbateľný oproti výhodám použitia fixnej veľkosti paketu, čo umožní hlavne menšiu zložitosť prepínačov. Preto sa aj experti z CCITT zhodli na použití paketov fixnej veľkosti, ktoré nazvali cell.

Veľkosť ATM cellu

Keď sme sa rozhodli pre pevnú veľkosť paketu, treba určiť jeho veľkosť. Zasa tu máme viacero faktorov, ktoré musíme brať do úvahy. Sú to efektívnosť prenosu, oneskorenie (ktoré vzniká pri vytváraní plných paketov ako aj pri ich čakaní vo frontách prepínačov) a zložitosť implementácie.

- **Efektívnosť prenosu**

Efektívnosť prenosu je daná pomerom medzi veľkosťou hlavičky a veľkosťou dátovej časti. Ak predpokladáme kompletne vyplnený paket, dostávame $\eta_H = \frac{L}{L+H}$. Teda čím väčšia je dátová časť, tým väčšiu efektívnosť dostávame.

- **Oneskorenie**

Niektoré parametre oneskorenia sú ovplyvňované veľkosťou paketu viac, iné menej. Medzi hlavné kategórie oneskorenia patria:

Oneskorenie pri napĺňaní paketov

Toto oneskorenie sa zväčšuje tým viac, čím väčšia je dátová časť paketu. Keďže táto veľkosť podstatne ovplyvňuje celkový výkon siete, môžu tu nastať problémy s jej časovou transparentnosťou, čo často vyžaduje použitie techník na rušenie ozvien hlavne pri prenose zvuku.

Oneskorenie pri prechode sieťou

Toto oneskorenie musí byť čo najmenšie hlavne pre hlasové prenosy. CCITT odporúča, že maximálne akceptovateľné oneskorenie by malo byť 24 ms (bez potreby rušičiek ozveny). Ak oneskorenie prekročí túto hodnotu, treba už použiť spomínané rušičky ozvien. Pri národných hovoroch možno dosiahnuť oneskorenie okolo 4 ms, avšak ak by sa prenos realizoval kombináciou ATM a inej siete, hodnotu 24 ms veľmi ľahko prekročíme.

Zoberme si príklad 1000 km prenosu s použitím 8 ATM prepínačov a 2 ATM-to-NonATM zariadení. Oneskorenie, ktoré nastane pre 32 bytové celly je okolo 14 ms, ale už pri použití 64 bytových cellov bude oneskorenie až 22 ms. Toto je dosť reálny prípad použitia, preto pri voľbe veľkých cellov by ATM nebolo vhodným médiom na prenos zvuku. Preto sa na výber veľkosti cellu môžeme pozrieť z viacerých pohľadov:

- Malý cell (32 a menej bytov), ktorý je vhodný skoro pre všetky zvukové prenosy
- Veľký cell (64 a viac bytov), kde môžeme uvažovať o dvoch prípadoch:
  - Použiť rušičky ozvien pre väčšinu zvukových prenosov.
  - Celly napĺňať iba sčasti, čím znížime oneskorenie pri napĺňaní paketov, ale to na úkor prenosovej efektívnosti. Na druhej strane zasa nepotrebujeme rušičky ozvien.

- Stredný cell (32 – 64 bytov). Použitím takýchto cellov sa môžeme vyhnúť rušičkám ozvien vo väčšine prípadov, kde počet ATM uzlov a prechodov na iné siete nie je príliš veľký. Tiež môžeme aplikovať čiastočné vypĺňanie cellov.

Oneskorenie vo frontách a pri depaketizácií

Oneskorenie vo frontách je ovplyvnené pomerom veľkostí dátovej časti L a hlavičky H. Ak zväčšíme veľkosť dátovej časti, zväčšíme aj toto oneskorenie. Na druhej strane ak dátovú časť zmenšíme, oneskorenie sa opäť zväčší, pretože vlastne relatívne zväčšíme veľkosť hlavičky, a preto čas potrebný na prenos nejakého množstva dát bude dlhší. Avšak celkový rozdiel medzi veľkosťami cellu 32 a 64 bytov je zanedbateľný, a to hlavne pri vysokej prevádzke na sieti (jedná sa asi o 40 μs).

Oneskorenie pri depaketizácií je akýmsi spojením oneskorení vo viacerých frontách. Je taktiež ovplyvňované veľkosťou cellu. Ak by celkové oneskorenie bolo príliš veľké, vyžaduje si to veľký buffer na strane príjemcu.

- **Implementačná zložitosť**

Táto je určená dvoma faktormi a to sú rýchlosť a počet cellov krát jeho veľkosť. Na garanciu zvoleného stratového pomeru musíme poznať počet cellov vzhľadom na frontu. Toto číslo nezávisí od veľkosti cellu, ale na druhej strane čím väčší cell máme, tým väčšiu pamäť na frontu musíme použiť. Zväčšenie cellu tiež zväčšuje aj čas potrebný na jeho spracovanie, a teda zmenšuje možnú dosiahnuteľnú prenosovú rýchlosť.

Napríklad ak zoberieme do úvahy rýchlosť 150 Mbit/s, frontu veľkosti 50 cellov a hlavičku veľkosti 4 byty, dospejeme k takýmto výsledkom:

- pre 16-bytový cell potrebujeme 8.000 bitov pamäte, ale na spracovanie cellu máme maximálne 1 μs.
- pre 256-bytový cell potrebujeme viac ako 64.000 bitov pamäte, a na spracovanie cellu máme už 15 μs.

Čas, ktorý sme zistili, však nie je rozhodujúci, pretože aj za 1 μs sa dá cell spracovať. Preto sa orientujeme hlavne na pamäťovú zložitosť.

- **Záver**

Na veľkosť cellu vplývajú protichodné faktory, ale najpreferovanejšie veľkosti sú medzi 32 a 64 bytov. Výber je teda ovplyvnený hlavne celkovým oneskorením, efektívnosťou prenosu a zložitosťou implementácie.

Pri konečnom rozhodnutí boli preferované veľkosti 32 bytov (Európa) a 64 bytov (USA a Japonsko). Výsledná veľkosť 48 bytov bola schválená aj v CCITT a bol to kompromis medzi oboma možnosťami.

Funkčnosť hlavičky (Kap. 2.4.5.4)

Virtuálne spojenia

ATM používa iba malú hlavičku. Ostatné potrebné informácie boli poskytnuté už pri vytvorení spojenia. Informácie ako adresa prijímateľa, sekvenčné číslo a podobne nie sú preto treba tak, ako je to nutné pri nespájaných sieťach. Každé virtuálne spojenie je identifikované číselným identifikátorom, ktorý má iba lokálny význam pre spojenie.

Mechanizmus na kontrolu toku cellov (*flow control*) ako napríklad ARQ (Automatic Repeat reQuest) ATM nepodporuje, pretože pri rýchlosti od 150 Mbit/s by sa prenieslo veľké množstvo dát, kým by sa dostavila reakcia, čo by spôsobilo nízku efektívnosť.

Hlavička teda musí identifikovať virtuálne spojenie. Na to sú v nej obsiahnuté dva identifikátory a to: VCI (*Virtual Channel Identifier*), ktorý identifikuje dynamicky alokované spojenia a VPI (*Virtual Path Identifier*), ktorý identifikuje staticky alokované spojenia.

Virtuálne kanály

Na identifikáciu virtuálneho kanálu v hlavičke slúži VCI. V budúcnosti sa budú vo veľkom používať optické káble, ktoré sú schopné prenášať stovky Mbit/s, ale virtuálne kanály budú často prenášať iba kbit/s. Preto po jednom optickom spojení môžu byť prenášané rádovo až niekoľko desaťtisícov virtuálnych kanálov. Toto vyžaduje až 16 bitový VCL identifikátor. Pretože ATM je založená na spojitej komunikácií, každé spojenie je charakterizované VCI a toto je vybraté pri vytváraní kanálu. VCI má iba lokálny význam pre spojenie medzi uzlami ATM. Po ukončení komunikácie niektorým kanálom sa uvoľní aj jeho VCI, ktorý bude opäť použiteľný.

Virtuálne cesty

Sú to polopermanentné spojenia medzi dvomi koncovými uzlami. Cez ne sa budú prepravovať viaceré simultánne spojenia (teda viaceré kanály). Takýto návrh umožňuje vytvárať virtuálne siete. Ďalšou výhodou takto alokovaných ciest je menšia náročnosť riadenia siete a sieťových zdrojov. Na vytvorenie takýchto virtuálnych sietí slúži VPI.

Manažment týchto virtuálnych ciest neprebieha iba nad jedným spojením, ale väčšinou je na úrovni zväzkov takýchto spojení. Preto VPI má 8 – 12 bitov, čo umožňuje 256 až 4096 virtuálnych ciest. Každá z týchto ciest môže obsahovať až 64 K virtuálnych kanálov určených pomocou VCI.

Priorita

Ďalšou funkciou, ktorú hlavička cellu zabezpečuje, je určenie rôznych priorít logických spojení. Priorita je požadovaná kvôli tomu, že pomocou nej sa dá sieť lepšie štrukturovať na logické podsiete. Priorita je užitočná aj pri vysokom zaťažení siete, kde môžeme v prípade potreby obetovať celly s nízkou prioritou.

Rozlišujeme dva druhy priorít, a to časová priorita a sémantická priorita. Časová priorita predpokladá, že niektoré celly môžu byť v sieti pozdržané dlhšie ako iné, teda pre niektoré spojenia sa vytvorí lepšia časová transparentnosť na úkor iných. Sémantická priorita zasa určuje celly, ktorých strata nie je až taká významná, teda sa tým definuje rôzna sémantická transparentnosť.

Prioritu môžeme priradiť či už celej virtuálnej ceste, kanálu prípadne aj samotným cellom. Prioritu možno definovať explicitne v hlavičke alebo implicitne pri nadviazaní virtuálneho spojenia.

Riešením s implicitnou prioritou môžeme definovať viacero úrovní priorít, ale na jeho implementáciu treba použiť tabuľky v prepínačoch, čo môže byť náročnejšie na implementáciu. Explicitné priority síce zaberú miesto v hlavičke, ale ich vyhodnocovanie bude oveľa jednoduchšie. Preferovaný je malý počet explicitných prioritných bitov (0 – 4 bity).

Údržba

Na podporu správy siete a na monitorovanie výkonu sú pridané do hlavičky ešte ďalšie informačné bity. Výhodné je použiť bit na rozlíšenie normálnych a údržbových dát. Táto technika sa volá PTI (Payload Type Identification). V takto odlíšených celloch môže sieť posielať rôzne informačné a testovacie dáta (napr. pre CRC, kvalitu spojenia).

Do cellu je vhodné umiestniť aj paritný bit, ktorým dokážeme monitorovať chyby (nepárneho počtu bytov). ATM používa v súčasnosti 0 – 2 údržbové bity.

Viacnásobný prístup

Ide vlastne o spojenie typu *point-to-multipoint*, keď z jednej fyzickej adresy možno dáta smerovať k viacerým koncovým užívateľom a to len za použitia toho istého spojenia. Na takúto funkciu opäť treba rozšíriť hlavičku cellu o nejaké ďalšie bity (0 – 8 bitov, čo závisí od MAC – *Medium Access Control*).

Ochrana hlavičky proti chybám

Ak nastane pri prenose hlavičky cellu chyba, toto môže mať negatívny vplyv na výkon siete, pretože zrejme nedorazí na správne miesto, ale môže nastať aj rozšírenie ďalších chýb. Preto je výhodné hlavičku doplniť o mechanizmus detekcie aspoň jednobitových chýb. Vhodným mechanizmom je použitie cyklického BCH kódu (Bose-Chadhuri-Hocquenghem). Napríklad ak chceme ochrániť 64bitový blok proti chybe jedného bitu, ktorú chceme vedieť aj opraviť, použijeme navyše 7 bitov. Tieto navyše ešte umožnia detekovať aj chyby väčšieho počtu bitov (ale už nie opraviť). Percentuálnu úspešnosť detekcie chýb a korekcie chýb jedného bitu je uvedená v nasledujúcej tabuľke.

# pridaných bitov → # chránených bitov ↓	6	7	8
32	48%	74%	89%
40	36%	68%	84%
48	23%	62%	81%

Hlavička ATM cellu je rozdelená podľa nasledovnej tabuľky. Špecifikácia podľa CCITT rozlišuje dva druhy hlavičiek, a to pre NNI (*Network Node Interface*) a pre UNI (*User Network Interface*).

funkcia	potrebné bity	CCITT (NNI / UNI)
VCI	8 – 16	16
VPI	8 – 12	12 / 8
Priorita	0 – 4	1
Údržba a typ cellu	0 – 2	2
Point-to-multipoint	0 – 8	0 / 4
HEC	0 – 8	8
RESERVED	0 – 6	1
SPOLU	16 – 56	40

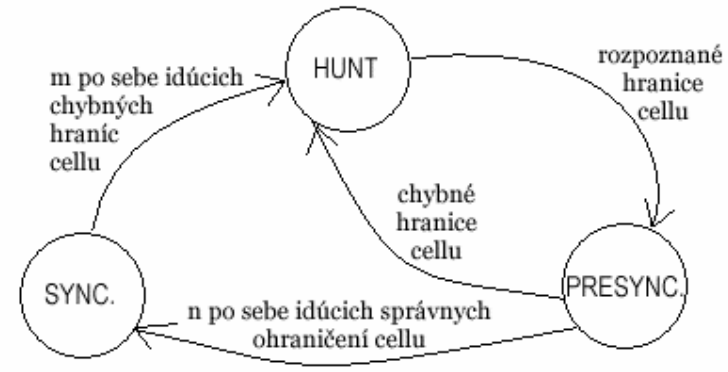
Podporné funkcie (Kap. 2.4.5.5)

Vo všetkých systémoch založených na prepínaní paketov musí byť prijímateľ schopný určiť hranice paketu. Napríklad HDLC, kde sú povolené pakety variabilnej dĺžky sa na označenie konca paketu používa istá vzorka bitov. Treba však zaručiť, aby sa nevyskytovala súčastne aj v pakete (to je zabezpečené technikou „bit stuffing“).

Vzorku bitov možno použiť aj v ATM, ale „bit stuffing“ je kvôli vysokým rýchlostiam ATM nevhodné. Avšak vďaka pevnej veľkosti paketov tu možno použiť namiesto „bit stuffing“ aj iné techniky. Ich podstatou je:

- Použiť prázdne celly
- Kontrolovať HEC (Header Error Code)
- Periodická synchronizácia cellov

Každé z týchto riešení možno opísať stavovým diagramom zo stavmi HUNT, PRESYNC a SYNC. V stave HUNT iba monitoruje linku a hľadá (napr. bit po bite), či nenájde nejaké hranice cellu. Ak ich nájde, prejde do stavu PRESYNC, kde kontroluje cell po celle ich hranice. Ak ich nájde *n*, potom sa prepne do stavu SYNC. Z tohto stavu odíde, iba keď stratí *m* po sebe nasledujúcich hraníc cellov. Parametre *n* a *m* určujú, ako rýchlo sa systém zosynchronizuje, a ako rýchlo odhalí opak.



Použitie prázdnych cellov

Prázdne celly sú charakterizované použitím špecifickej hodnoty v hlavičke. V stave HUT hľadáme vo vstupných dátach špecifickú hodnotu hlavičky. Treba zabezpečiť, aby sa táto postupnosť nevyskytovala v ostatných regulárnych dátach.

V stave PRESYNC sa opäť hľadá táto hodnota na potvrdenie hraníc cellu. Pretože prázdne celly prichádzajú po linke iba keď neprebieha žiadna iná komunikácia, niekedy sa bude na ne musieť čakať dlhšie. Podobná situácia je aj v stave SYNC.

Kontrola HEC

Táto metóda využíva vzájomný vzťah medzi bitmi hlavičky a nasledovnými bitmi cyklického HEC kódu.

V stave PRESYNC kontrolujeme bit po bite blok, ktorý by vyhovoval pravidlám HEC kódu. Ak nevyhovuje, presunieme sa o bit ďalej. Ak takýto blok nájdeme, prepneme sa do PRESYNC, kde potom ich rátame až do *n* takýchto blokov a prepneme sa do SYNC. Zo stavu SYNC odchádzame po *m* chybných HEC kódoch do stavu HUNT.

Problémom je, keď sa blok, ktorý spĺňa HEC kód nachádza aj v dátovej časti cellu. Vtedy ľahko stratíme synchronizáciu, prípadne ju ťažšie obnovíme. Preto sa používa technika scramblovania, ktorá zabezpečí, že dátový blok je pseudonáhodnou postupnosťou, čím sa zníži pravdepodobnosť výskytu vzorky, ktorá by bola aj HEC kódom.

Periodická synchronizácia cellov

Môže sa stať, že predchádzajúce riešenia z nejakých dôvodov nevyhovujú. Ďalšou možnosťou je použiť periodickú synchronizáciu cellov, ktorá je založená na periodickom vkladaní synchronizačných cellov do dátového toku. Takýto cell je charakterizovaný napríklad špeciálnou hlavičkou. Tento druh synchronizácie je vlastne špeciálnym prípadom prvého riešenia, s tým rozdielom, že namiesto posielania prázdnych cellov použijeme celly so špeciálnou hlavičkou. Opäť sa tu klasicky aplikuje prechod stavmi HUNT → PRESYNC. → SYNC. a späť. Toto riešenie nie je nepriaznivo ovplyvňované ani vkladaním „rušivých“

blokov dát – teda prázdnych hlavičiek do dátového bloku, pretože systém pravidelne posiela synchronizačné celly automaticky. Perióda T je zvolená ako kompromis medzi efektívnosťou prenosu a potrebného synchronizačného času. Malá perióda znižuje efektívnosť, ale zmenšuje čas potrebný na synchronizáciu.

Záver

CCITT nakoniec vybrala druhé riešenie, t.j. synchronizácia založená na HEC a scrambling informačného bloku.