ЛЕКЦІЯ 4

Тема: ТЕХНОЛОГІЇ ВІДКРИТИХ СИСТЕМ

План

1. Основні поняття відкритих систем
2. Історія розвитку технології відкритих систем
3. Еталонна модель взаємодії відкритих систем
4. Характеристика рівнів моделі взаємодії відкритих систем

1. Основні поняття відкритих систем

Одним з основних напрямків інформаційних технологій, що визначає ефективність функціонування економічних об'єктів, виступає технологія відкритих систем. Ідеологію відкритих систем реалізують в своїх останніх розробках всі провідні фірми-постачальники засобів обчислювальної техніки, передачі інформації і програмного забезпечення. Їх результативність на ринку інформаційних технологій і систем визначається узгодженої науково-технічної політикою і реалізацією стандартів відкритих систем.

Існують декілька визначень поняття «відкрита система», які в різний час були сформульовані такими організаціями, як Інститут інженерів з електротехніки й електроніки (Institute of Electrical and Electronics Engineers, IEEE), Національний інститут зі стандартів і технологій США (National Institute of Standards and Technology, NIST), компанією Hewlett-Packard. У широкому значенні *відкритою системою* може бути названа будь-яка система (телекомунікаційна мережа, автоматична телефонна станція, абонентський термінал, операційна система, протокол, інші апаратні й програмні продукти), побудована згідно з відкритими специфікаціями. Відповідно до визначення IEEE, «відкрита система — це система, що реалізує відкриті специфікації на інтерфейси, служби й формати даних, достатні для того, щоб забезпечити:

• можливість перенесення прикладних програм, розроблених належним чином з мінімальними змінами, на широкий діапазон систем;

• спільну роботу з іншими прикладними системами на локальних і віддалених платформах;

• взаємодія з користувачами в стилі, що полегшує перехід від системи до системи».

Відкриті системи мають наступні властивості, які представлені на рис. 1

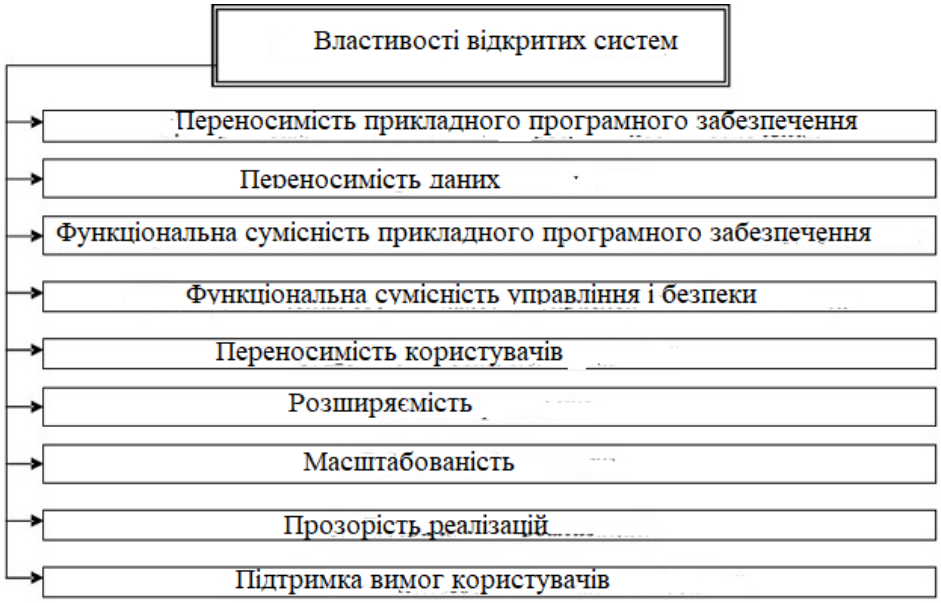

Рис. 1 - Властивості відкритих систем

1. **Переносимість прикладного програмного забезпечення** та повторна застосованість програмного забезпечення. Під переносимістю додатків розуміється перенесення всього відповідного даному додатку програмного забезпечення на інші платформи. Під повторним використанням програмного забезпечення розуміється перенесення в нові додатки деякої частини працюючих програм, що також має велике практичне значення і безпосередньо відноситься до цілей відкритості систем.
2. **Переносимість даних** означає можливість перенесення на нові прикладні платформи даних, що зберігаються в зовнішній пам'яті існуючих систем інформаційних технологій. Переносимість даних забезпечується застосуванням у відкритих системах стандартів, суворо регламентують формати і способи представлення даних.
3. **Функціональна сумісність** (інтероперабельність) прикладного програмного забезпечення - це можливість обміну даними між різними прикладними програмами, в тому числі між програмами, реалізованими на різнорідних прикладних платформах, а також можливість спільного використання даних.
4. **Функціональна сумісність (інтероперабельність) управління і безпеки** - це уніфікація і цілісність засобів адміністративного управління та управління інформаційною безпекою, тобто для забезпечення інтеграції систем, їх кошти адміністративного управління і засоби захисту повинні будуватися відповідно до міжнародних стандартів.
5. **Переносимість користувачів** - це забезпечення можливості для користувачів інформаційних технологій уникнути необхідності перенавчання при взаємодії з системами, реалізованими на основі різних платформ.
6. **Можливість розширення** - це здатність системи еволюціонувати з урахуванням змін стандартів, технологій і призначених для користувача вимог.
7. **Масштабованість** - властивість системи, що дозволяє їй ефективно працювати в широкому діапазоні параметрів, що визначають технічні і ресурсні характеристики системи (прикладами таких характеристик можуть служити:

число процесорів, число вузлів мережі, максимальне число обслуговуваних користувачів).

8. **Прозорість реалізацій** - це спосіб побудови системи, при якому всі особливості її реалізації ховаються за стандартними інтерфейсами, що і забезпечує властивість прозорості реалізацій інформаційних технологій для кінцевих користувачів систем.

9. **Підтримка призначених для користувача вимог** - це точна специфікація призначених для користувача вимог, визначених у вигляді наборів сервісів, що надаються відкритими системами додатків користувачів.

Однак відкрита система необов'язково повинна бути повністю доступна іншим відкритим системам. Це обмеження може бути викликано необхідністю захисту інформації в комп'ютерах і засобах комунікацій і забезпечується шляхом фізичного відділення або шляхом використання технічних можливостей. Сутність технології відкритих систем полягає в забезпеченні можливості переносимості прикладних програм між різними платформами і взаємодії систем між собою. Ця можливість досягається за рахунок використання міжнародних стандартів на всі програмні та апаратні інтерфейси між компонентами систем.

Стандарти прагнуть зайняти центральне місце в напрямку розвитку відкритих систем і в індустрії інформаційних технологій. Більше 250 підкомітетів в офіційних організаціях по стандартизації і уніфікації працюють над стандартами в області інформаційних технологій. Більше 1000 стандартів або вже прийнято цими організаціями, або знаходяться в процесі розробки.
При цьому розрізняють стандарти де-факто і де-юре, представлені на рис. 2

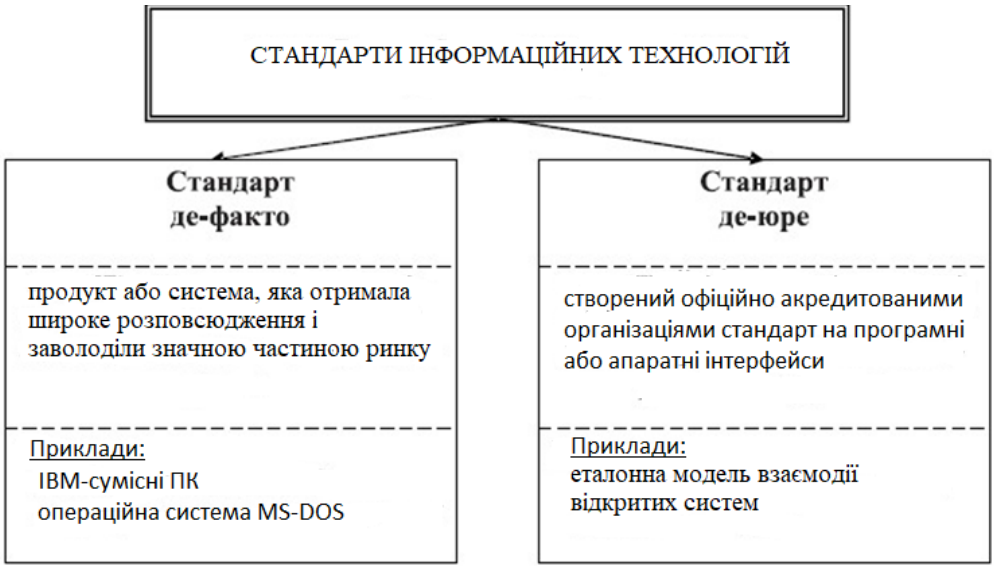

Рис. 2 - Види стандартів інформаційних технологій

Стандарт де-факто означає, що продукт або система якогось конкретного виробника захопили значну частину ринку і інші виробники прагнуть емулювати, копіювати або використовувати їх з тим, щоб розширити свій сектор ринку.

Стандарт де-юре створюється офіційно акредитованими організаціями по розробці стандартів. Він розробляється за правилами досягнення угоди у відкритому обговоренні, в якому може взяти участь кожен бажаючий. При створенні промислових стандартів жодна з груп не може діяти незалежно. Якщо одна котрась із груп виробників створює стандарт, в якому не потребують користувачі, вона зазнає невдачі. Те ж саме можна сказати і про зворотний випадок, коли користувачі створять стандарт, з яким виробники не зможуть або не захочуть погодитися, - спроба створення такого стандарту також буде безуспішною.

Технологія відкритих систем користується успіхом тому, що забезпечує переваги для різного роду фахівців, пов'язаних з областю інформаційних технологій.

Для користувача відкриті системи забезпечують:	• нові можливості збереження зроблених вкладень завдяки властивостям еволюції, поступового розвитку функцій систем, заміни окремих компонентів без перебудови всієї системи; • звільнення від залежності від одного постачальника апаратних або програмних засобів, можливість вибору продуктів із запропонованих на ринку за умови дотримання постачальником відповідних стандартів відкритих систем; • дружність середовища, в якій працює користувач, мобільність персоналу в процесі еволюції системи; • можливість використання інформаційний ресурсів, наявних в інших системах (організаціях).
Проектувальник інформаційних систем отримує:	• можливість використання різних апаратних платформ; • можливість спільного використання прикладних програм, реалізованих в різних операційних системах; • розвинені засоби інструментальних середовищ, які підтримують проектування; • можливості використання готових програмних продуктів і інформаційних ресурсів.
Розробники загальносистемних програмних засобів мають:	• нові можливості поділу праці, завдяки повторному використанню програм; • розвинені інструментальні середовища і системи програмування; • можливості модульної організації програмних комплексів, завдяки стандартизації програмних інтерфейсів

Модульна організація програмних комплексів, завдяки стандартизації програмних інтерфейсів, дозволяє переглянути дублювання функцій в різних програмних продуктах, через що системи, що інтегрують ці продукти, непомірно розростаються за обсягом, втрачають ефективність. Відомо, що в тій же області обробки даних і текстів багато продуктів, що пропонуються на ринку (текстові редактори, настільні видавничі системи, електронні таблиці, системи управління базами даних) по ряду функцій дублюють один одного, а іноді і підміняють функції операційних систем. Крім того, помічено, що в кожній новій версії цих продуктів розміри їх збільшуються на 15%.

У розподілених системах, що містять кілька робочих місць на персональних комп'ютерах і серверах в локальній мережі, надмірність програмних кодів через дублювання зростає багаторазово. Ідеологія і стандарти відкритих систем дозволяють по-новому поглянути на розподіл функцій між програмними компонентами систем і тим самим значно підвищити ефективність.

2 Історія розвитку технології відкритих систем

Потреба в застосуванні відкритих систем виникла ще на зорі використання обчислювальної техніки. Вона була обумовлена декількома причинами:

1. Для вирішення все більш широкого діапазону завдань створювалися програми, які вимагали створення різноманітних апаратних платформ, які виконують ці програми. У свою чергу, впровадження неоднорідних систем і бажання розділяти між такими системами інформацію привели до необхідності забезпечити можливість їх спільної роботи.
2. Розробники програмних додатків були зацікавлені в скороченні витрат і часу перенесення своїх додатків на різні платформи, а для цього була потрібна сумісність між різними апаратними платформами.
3. Виробники апаратних платформ були зацікавлені в створенні таких систем, які здатні виконувати широкий діапазон існуючих прикладних програмних додатків, а для цього також необхідно було розробити стандарти їх сумісності.

Необхідність вирішення цих проблем поступово привела до створення концепції відкритих систем.

Таблиця 1. Етапи розвитку технології відкритих систем

Етап	**Опис етапу**	**Характеристика етапу**
1-й етап	Створення IBM 360	З'явилася програмна сумісність між моделями одного сімейства; З'явилася можливість об'єднання декількох машин в одну обчислювальну систему
2-й етап	Розробка стандартів мов програмування	• Стандартизовані мови програмування
3-й етап	Створення суперміні-EOM VAX	• EOM цього сімейства стали стандартною платформою для розробки систем проектування, систем збору і обробки даних і т. д.

4-й этап	Розробка моделі взаємозв'язку відкритих систем	• Міжнародна організація стандартизації розробила загальні принципи взаємозв'язку відкритих систем
5-й этап	Поява операційної системи MS-DOS	• Було розроблено величезну кількість прикладних програм для персональних комп'ютерів, що працювали під управлінням операційної системи MS-DOS і сумісних з нею систем
6-й этап	Поява процесора з архітектурою RISC	• З'явилася апаратна база для реалізації ефективної переносимості програм, написаних на мовах високого рівня, для процесорів різних виробників
7-й этап	Впровадження операційної системи UNIX	• Операційна система UNIX забезпечує високу переносимість створюваних для роботи в ній прикладних програм в інші системи

1-й етап починається з того моменту, коли виникла проблема переносимості програм і даних між комп'ютерами з різною архітектурою. Одним з перших кроків в цьому напрямку стало створення в 1964 р шести моделей сімейства IBM 360, що стали першими комп'ютерами третього покоління. Моделі мали єдину систему команд і відрізнялися один від одного об'ємом оперативної пам'яті і продуктивністю. При створенні моделей сімейства використовувався ряд нових принципів, що робило машини універсальними і дозволяло з однаковою ефективністю застосовувати їх як для вирішення завдань в різних областях науки і техніки, так і для обробки даних в сфері управління і бізнесу.

Найбільш важливими нововведеннями ЕОМ цього сімейства є:

• програмна сумісність всіх моделей сімейства;

• можливість підключення великої кількості зовнішніх пристроїв і стандартного сполучення цих пристроїв з процесором через апаратуру каналів зв'язку (малася можливість об'єднати кілька машин в одну обчислювальну систему для вирішення різного виду завдань).

2-й етап. Часткове вирішення проблеми мобільності для програм і програмістів забезпечили ранні стандарти мов програмування, наприклад, Фортран і Кобола. Мови дозволяли створювати стерпні програми, хоча часто і обмежували функціональні можливості. Мобільність забезпечувалася також і за рахунок того, що ці стандарти були прийняті багатьма виробниками різних платформ. Коли мови набували статус стандарту, їх розробкою і супроводом починали займатися національні та міжнародні організації зі стандартизації. Досягнення цього рівня мобільності було першим прикладом справжніх можливостей відкритих систем.

3-й етап у розвитку технології відкритих систем - це друга половина 70-х рр. XX ст. Він пов'язаний з областю інтерактивної обробки і збільшенням обсягу продуктів, для яких потрібно переносимість, наприклад пакети для інженерної графіки, системи автоматизованого проектування (САПР), системи управління базами даних (СКБД). У цей час фірма DIGITAL почала випуск суперміні-ЕОМ VAX. ЕОМ цієї серії мали 32- архітектуру, а програмісти

отримали можливість прямо використовувати адресний простір значного обсягу, що практично знімало всі обмеження на розміри вирішуваних завдань. Машини цього типу надовго стали стандартною платформою для систем проектування, збору і обробки даних, обслуговування експерименту і т. П.

4-й етап відноситься до кінця 70-х років і пов'язаний з розвитком мережевих технологій. В цей час комп'ютерні мережі, що використовують протоколи INTERNET, почали широко застосовуватися для об'єднання систем військових і академічних організацій США. Паралельно компанія IBM розробила і стала застосовувати власну мережеву архітектуру. Коли мережева обробка стала реальністю, користувачі почали звертати увагу на сумісність і можливість інтеграції як на необхідні атрибути відкритих систем. Міжнародна організація стандартизації ISO в 1977-78 роках розгорнула інтенсивні роботи зі створення стандартів взаємозв'язку в мережах відкритих систем. В ході цих робіт була створена семиуровневая модель взаємозв'язку відкритих систем OSI - Open Systems Interconnection Basic Reference Model.

Модель взаємозв'язку відкритих систем описує загальні принципи взаємозв'язку відкритих систем і використовується в якості основи для розробки стандартів ISO. Тоді ж вперше було введено визначення відкритої інформаційної системи.

В цей же час були зроблені перші системи, які забезпечували організацію використання розподілених ресурсів в системі. Реалізована фірмою DIGITAL EQUIPMENT система забезпечила об'єднання декількох десятків суперміні-ЕОМ VAX за допомогою спеціальної високо швидкісної лінії зв'язку і локальної мережі Ethernet. У цій системі з'явилася можливість вирішувати завдання поділу ресурсів (пам'яті, процесорів, баз даних і т. П.).

5-й етап (перша половина 80-х рр.) Характеризується масовим поширенням персональних комп'ютерів з операційною системою MS-DOS корпорації Microsoft. Низька ціна і широке поширення створили величезний ринок для даної операційної системи і прикладних програм, написаних для неї. Багато прикладні програми, що виконуються в MS-DOS, можуть виконуватися і на будь-який інший сумісної системі. Але ця сумісність обмежена архітектурою з 16-розрядної адресацією, графікою низького дозволу і неможливістю виконувати більш одного завдання одночасно. Для середовища MS-DOS характерний також ризик швидкого поширення вірусів, оскільки система слабо захищена на програмному та апаратному рівнях.

6-й етап пов'язаний зі створенням першого RISC-процесора в 1982 р Це подія не викликала в той час великих відгуків, проте воно в значній мірі визначило розвиток відкритих систем до кінця десятиліття і грає вирішальну роль і сьогодні. По-перше, RISC-архітектура забезпечила істотне підвищення продуктивності мікропроцесорів, а по-друге, надала апаратну базу для реалізації ефективної переносимості програм для процесорів різних виробників.

Характерна для архітектури RISC-елементарність набору команд дозволяє наблизити ефективність програм, написаних на мовах високого рівня, до ефективності програм в машинному коді і автоматизувати процес налаштування програм для їх оптимізації. В результаті стало можливим забезпечити на рівні мов високого рівня ефективну мобільність програм.

7-й етап у розвитку технології відкритих систем пов'язаний з впровадженням операційної системи UNIX. Хоча ОС UNIX була розроблена до створення MS-DOS, вона не могла ефективно використовуватися, так як вимагала значних апаратних ресурсів. З появою потужних RISC-мікропроцесорів UNIX проявила себе як найбільш перспективне відкрите операційне середовище. Історично ця операційна система виявилася найбільш життєвим варіантом для створення загальної бази переносимості. Вона задовольняє більшості вимог, що пред'являються до відкритих систем. Прикладні програми, що створюються для роботи в UNIX, за певних умов можуть мати досить високу переносимість як в інші UNIX-подібні системи, так і в системи, що задовольняють стандартам на розроблені інтерфейси.

Міжнародні стандарти повинні бути реалізовані для кожного системного компонента мережі, включаючи кожну операційну систему і прикладні пакети. До тих пір, поки компоненти задовольняють таким стандартам, вони відповідають цілям відкритих систем.

3 Еталонна модель взаємодії відкритих систем

Переміщення інформації між комп'ютерами різної конфігурації є надзвичайно складним завданням. На початку 1980-х рр. Міжнародна організація стандартизації (ISO) і Міжнародний консультативний комітет з телеграфії і телефонії (МККТТ) визнали необхідність в створенні моделі мережі, яка могла б допомогти постачальникам створювати реалізації взаємодіючих мереж.

Передумовами розробки моделей взаємодії відкритих систем з'явилися:

• необхідність еталонної системи, яка допоможе забезпечити взаємодію мережевих засобів, що пропонуються різними розробниками;

• необхідність теоретично обґрунтованої мережевої моделі, яка вирішує завдання переміщення інформації між комп'ютерами різних систем;

• розбиття загальної задачі переміщення інформації на більш дрібні підзадачі, що дозволило б розробникам мережевих додатків сконцентруватися на вирішенні конкретних прикладних задач.

У 1984 р Міжнародна організація стандартизації розробила еталонну модель мережі під назвою "Взаємодія відкритих систем" (OSI - Open System Interconnection).

Взаємодія двох додатків за допомогою мережі є досить складним завданням, яке включає в себе:

1. Пошук додатків, з якими буде проводитися обмін інформацією.
2. Встановлення і підтримка зв'язку.
3. Обробка втрат і перешкод при обміні.

Якби реалізація всіх необхідних складових для обміну інформацією лежала б тільки на додатках, то створення останніх було б вкрай складним завданням. Крім того, виникла б проблема узгодження транспортних засобів для додатків, випущених різними розробниками.

Модель взаємодії відкритих систем розділяє завдання мережевого обміну на сім дрібніших завдань, що спрощує рішення. Кожна з підзадач сформульована таким чином, щоб для її вирішення був потрібний мінімум

зовнішньої інформації. Кожен рівень моделі взаємодії відкритих систем відповідає своїй підзадачі, а значить, кожен рівень моделі в достатній мірі автономний. Функціонально рівні взаємодіють на строго ієрархічній основі: кожен рівень забезпечує сервіс для вищого рівня, запитуючи, в свою чергу, сервіс у нижчестоящого рівня.

До основних принципів розробки мережевих рівнів, що відповідають моделі взаємодії відкритих систем, відносяться:

1. Кожен рівень повинен виконувати строго певну функцію.

2. Набір функцій, які виконуються мережевим рівнем, приводиться у відповідність із загальноприйнятими міжнародними стандартами.

3. Межі рівня вибираються таким чином, щоб мінімізувати потік даних, який проходить через них.

4. Кількість мережевих рівнів повинно бути достатньо великим, щоб не розміщувати різні функції на одному і тому ж рівні і в той же час не ускладнювати модель, роблячи її неосяжної.

З появою в 1984 р. першого стандарту по еталонній моделі і, особливо, з подальшим наповненням цієї моделі конкретними протоколами практична привабливість моделі взаємодії відкритих систем як єдиного комплексу стандартів, що реалізують взаємну сумісність обладнання та програм різних постачальників, почала сильно зростати. Свідчення тому - поява на початку 90-х рр. урядових профілів взаємодії відкритих систем (GOSIP - Government Open Systems Interconnection Profile) майже у всіх розвинених країнах світу, прийнятих і стандартизованих на державному рівні, а також спроби об'єднання GOSIP різних країн в єдину Промислово-урядову специфікацію відкритих систем.

Технологія передачі інформації в моделі взаємодії відкритих систем

Процес передачі даних з прикладної програми однієї системи в прикладну програму іншої системи, за умови, що обидві системи задовольняють стандартам еталонної моделі взаємодії відкритих систем і мають семирівневу структуру, представлений на рис. 3.

1-й етап. Прикладна програма, яка є джерелом інформації, передає дані верхнього рівня системи, в середовищі якої вона реалізована. На цьому рівні відбувається обробка отриманих даних, сенс якої полягає в тому, що до інформації додається заголовок, що містить службову інформацію, необхідну для адресації повідомлення і виконання контрольних функцій.

Керуюча інформація у формі кодованого заголовка поміщається перед фактичними даними, які повинні бути передані. Цей інформаційний блок передається в наступний суміжний нижчий рівень системи.

2-й етап. Кожен рівень системи, приймаючи інформацію від верхнього рівня, додає до неї свої дані, які необхідні для функціонування цього рівня. При проходженні чергового рівня зверху вниз дані отримують новий заголовок. Крім того, у міру просування через рівні, інформація кодується, поступово перетворюючись в сигнали, які можна передавати по каналах зв'язку.

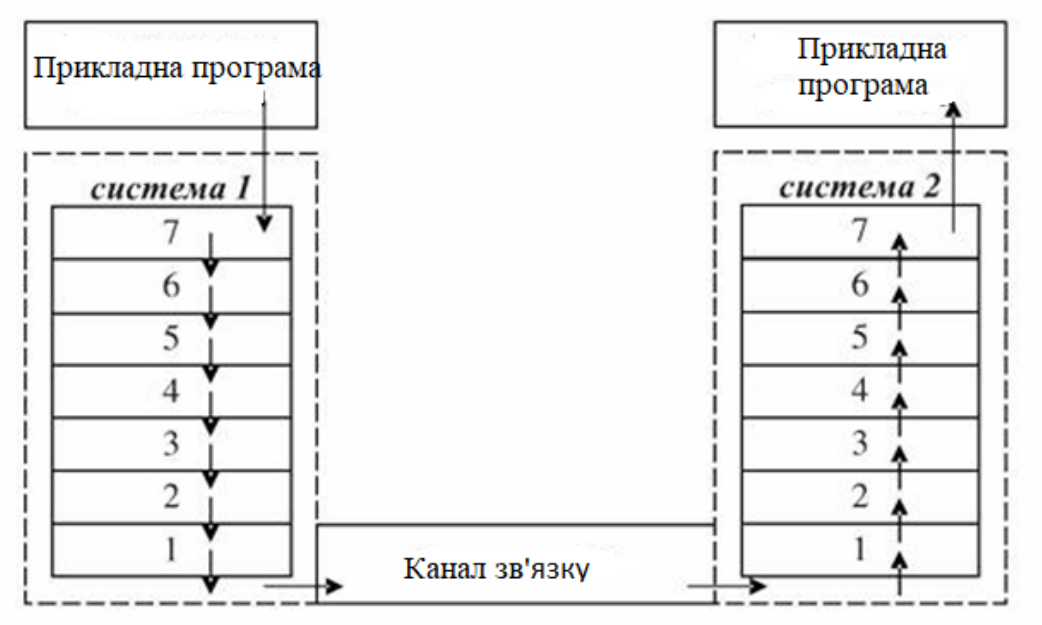

Рис. 5.3. Взаємодія відкритих систем на базі еталонної моделі

3-й етап. Нижній рівень системи, заголовка до повідомлення не додає, його функції полягають у передачі інформації, представленої у вигляді послідовності електричних сигналів, по каналу зв'язку.

4-й етап. На приймаючій стороні інформація проходить від низу до верху, і на кожному рівні відповідний заголовок повідомлення відділяється, тому рівень на приймаючій стороні отримує дані точно в тому вигляді, в якому вони були відправлені відповідним рівнем на протилежному боці. Отриманий заголовок прочитується, після чого відбувається обробка інформації відповідно до команд, що містяться в цьому заголовку.

5-й етап. Верхній рівень приймаючої системи передає дані прикладній програмі, з якою проводиться обмін інформацією, при цьому інформаційний блок містить тільки оригінальний текст, оскільки всі заголовки до цього моменту вже відділені від повідомлення.

Слід зазначити, що концепція заголовка і власне даних відносна і залежить від перспективи того рівня, який в даний момент аналізує інформаційний блок. Не всі рівні потребують приєднання заголовків, деякі просто виконують трансформацію фактичних даних, які вони отримують, щоб зробити їх більш-менш читаємими для суміжних з ними рівнів. Обробка повідомлення рівнями моделі взаємодії відкритих систем представлена на рис.4.

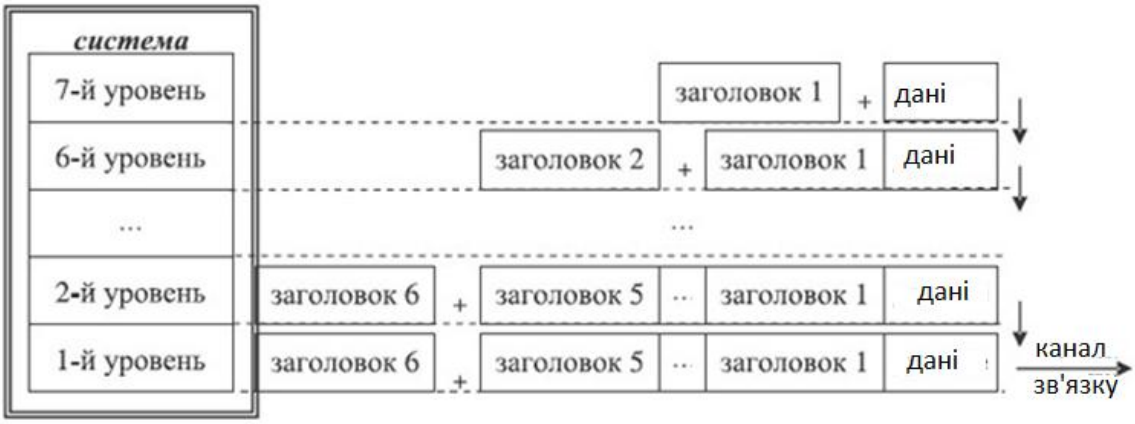

Рис. 4 - Технологія обробки повідомлення рівнями моделі взаємодії відкритих систем

Модель взаємодії відкритих систем описує тільки системні засоби взаємодії, не торкаючись додатків кінцевих користувачів. Додатки реалізують свої власні протоколи взаємодії, звертаючись до системних засобів.

Слід мати на увазі, що додаток може взяти на себе функції деяких верхніх рівнів моделі, в такому випадку, при необхідності мережевого обміну воно звертається безпосередньо до системних засобів, які виконують функції нижніх рівнів моделі взаємодії відкритих систем, що залишилися.

До заслуг еталонної моделі взаємодії відкритих систем можна віднести наступні:

1. Концепція рівневої архітектури взаємодії відкритих систем, закладені в ній принципи автоматичного узгодження параметрів різних рівнів, принципи побудови профілів і функціональних стандартів, протоколи окремих рівнів стали еталоном при вирішенні подібних питань у багатьох інших мережевих архітектур.
2. Багато з розроблених протоколів моделі, які безпосередньо не отримали широкого практичного застосування, послужили прямою основою для створення аналогічних протоколів інших мережевих архітектур, в тому числі в мережі Internet.
3. Багато стандартів, розроблені для еталонної моделі взаємодії відкритих систем, наприклад, стандарти по кодам, механічними параметрами з'єднувачів на фізичному рівні, мов програмування та ін., реалізовані в безлічі виробів різних фірм.

Однак, незважаючи на всі переваги еталонної моделі, їй притаманні і певні недоліки:

1. Достаток стандартів взаємодії відкритих систем.
2. Складність протоколів взаємодії відкритих систем і, як наслідок, порівняно висока вартість пристроїв, що реалізують ці протоколи.

3. Повільний процес розробки стандартів.
4. Слабке впровадження реальних комерційних продуктів або діючих систем.

Слід також мати на увазі, що комплект протоколів Internet досить міцно укорінився ще до того, як був розроблений досить працездатний комплект протоколів взаємодії відкритих систем, і якщо навіть деякі протоколи еталонної моделі перевершили потім за своїми функціональними можливостями і гнучкості відповідні протоколи Internet, всю встановлену базу Internet замінювати було пізно. До того ж концепція еталонної моделі, розроблена до того, як був закоріненим Internet, не передбачила чіткого плану переходу на інші технології і співіснування з ними.

4 Характеристика рівнів моделі взаємодії відкритих систем

Еталонна модель взаємодії відкритих систем складається з семи рівнів, представлених на рис.5.

Рис. 5 - Рівні еталонної моделі взаємодії відкритих систем

Функції всіх рівнів моделі взаємодії відкритих систем можуть бути віднесені до однієї з двох груп:

- Функції, які залежать від конкретної технічної реалізації мережі
- Функції, орієнтовані на роботу з додатками

Три нижніх рівні - фізичний, канальний і іноді мережевий - є мережезалежними, тобто протоколи цих рівнів тісно пов'язані з технічною реалізацією мережі, з комунікаційним обладнанням, тобто перехід на інше обладнання мережі означає повну зміну протоколів фізичного і канального рівня у всіх вузлах мережі.

Три верхніх рівня - сеансовий, рівень представлення і прикладний - орієнтовані на додатки і мало залежать від технічних особливостей побудови

мережі і є мереженезалежні. На протоколи цих рівнів не впливають ніякі зміни в топології мережі, заміна обладнання або перехід на іншу мережеву технологію.

Транспортний і мережевий рівні є проміжними, вони приховують всі деталі функціонування нижніх рівнів від верхніх рівнів. Це дозволяє розробляти додатки, які не залежать від технічних засобів, які безпосередньо займаються транспортуванням повідомлень.

Призначення і основні функції рівнів моделі взаємодії відкритих систем наведені в табл. 2.

Таблиця 2 – Призначення і основні функції рівнів еталонної моделі взаємодії відкритих систем

Рівень моделі	**Призначення рівня**	**Основні функції рівня**
Фізичний	Встановлення, підтримка роз'єднання фізичного каналу	Визначення характеристик фізичного ісередовища передачі даних Визначення характеристик електричних сигналів Передача послідовностей бітів
Канальний	Управління доступом до передавальної середовищі і управління передачею даних	Прийом пакетів, що надходять з мережевого рівня Підготовка пакетів до передачі Організація початку передачі інформації Передача інформації по каналу Перевірка одержуваної інформації та виправлення помилок Переведення каналу в пасивний стан
Мережевий	Прокладка оптимальних маршрутів для передачі пакетів даних через топологію підмереж зв'язку	Забезпечення незалежності передачі даних від використовуваних засобів передачі Управління швидкістю передачі блоків даних Вибір маршруту передачі і комутація (ретрансляція) даних Виявлення та виправлення помилок передачі даних
Транспортний	Забезпечення надійного, послідовного обміну даними між користувачами з використанням мережевого рівня	Розподіл довгих повідомлень, що надходять від верхніх рівнів, на пакети даних Управління темпом обміну Формування початкових повідомлень з набору пакетів, отриманих через нижні рівні Визначення якості сервісу, яке потрібно забезпечити за допомогою мережевого рівня, включаючи виявлення та усунення помилок

Сеансовий	Управління діалогом і надання засоба синхронізації	Вибір режиму передачі між прикладними процесами Управління черговістю передачі даних і їх пріоритетом Визначення точки синхронізації Здійснення повторної установки сеансового з'єднання в заздалегідь певний стан за запитом представницького рівня Відновлення сеансу
Представницький	Забезпечення незалежності прикладних об'єктів від використання конкретного синтаксису (кодування) інформації, що передається	Запит встановлення сеансу Вибір правил кодування інформації Узгодження і повторне погодження правил кодування інформації Шифрування і дешифрування даних для забезпечення секретності обміну даними для всіх прикладних служб Запит завершення сеансу
Прикладний	Забезпечення доступу прикладних процесів до середовища передачі інформації для забезпечення їх взаємодії при вирішенні загальної задачі	Ідентифікація партнерів, які передбачають взаємодіяти Встановлення повноважень для передачі Узгодження механізму секретності Передача прикладних даних Узгодження відповідальності за виявлення помилок і процедур управління цілісністю даних Ідентифікація обмежень по синтаксису даних (безліч символів, структури даних)

Фізичний рівень - базовий рівень в ієрархії протоколів моделі взаємодії відкритих систем.

Призначення фізичного рівня полягає в забезпеченні механічних, електричних, функціональних і процедурних засобів для встановлення, підтримки і роз'єднання фізичних з'єднань з метою передачі послідовностей бітів між об'єктами мережі.

Функції фізичного рівня:

• встановлення та роз'єднання фізичних з'єднань між об'єктами комп'ютерної мережі;

• визначення характеристик фізичного середовища передачі даних, таких як смуга пропускання, перешкодозахищеність та ін .;

• визначення характеристик електричних сигналів, таких, як рівень напруги або струму сигналу, що передається, тип кодування, швидкість передачі сигналів;

• передача послідовностей бітів в синхронному або асинхронному режимі.

У свою чергу, фізичний рівень надає канальному рівню наступні послуги:

• фізичні з'єднання;
• ідентифікацію фізичних каналів передачі даних;
• організацію передачі послідовностей бітів;
• оповіщення про несправності фізичного рівня;
• визначення параметрів якості послуг, що надаються.

Фізичний рівень отримує пакети даних від вищого канального рівня, перетворює їх в оптичні або електричні сигнали, відповідні 0 і 1 бінарного потоку і забезпечує перенесення потоку двійкових сигналів, у вигляді яких представляються дані, що передаються, через фізичне середовище, що з'єднує об'єкти комп'ютерної мережі.

На физичному рівні дані передаються двома способами: **аналоговим і цифровим.** Як фізичне середовище, як правило, використовується мережа комутованих каналів зв'язку, що з'єднують кореспондуючі пари об'єктів мережі. При передачі даних по аналоговим каналам зв'язку послідовність біт на вході каналу перетвориться в пристроях модуляції / демодуляції - модемах в аналогові сигнали, параметри яких узгоджені з параметрами фізичного середовища. Прийняті на виході аналогового каналу сигнали назад перетворюються в послідовність біт. У разі використання цифрових каналів зв'язку перетворення послідовностей бітів в аналогові сигнали не проводиться. При цьому замість модемів використовують лінійні контролери, які забезпечують з'єднання обладнання обробки даних з фізичним каналом.

Фізичний рівень найменш суперечливий, його функції реалізовані тільки апаратними засобами, причому на апаратуру розроблені і широко використовуються міжнародні стандарти. Функції фізичного рівня реалізуються у всіх пристроях, підключених до мережі. З боку комп'ютера функції фізичного рівня виконуються кінцевими активними мережевими пристроями - мережевою картою і модемом.

Канальний рівень забезпечує надійну передачу масивів даних між мережевими відкритими системами, безпосередньо пов'язаними деяким фізичним середовищем передачі даних.

Призначення канального рівня полягає в управлінні доступом до передавального середовища і в управлінні передачею даних.

Функції канального рівня:
• прийом пакетів, що надходять з мережевого рівня;
• підготовка пакетів до передачі;
• генерація стартового сигналу і організація початку передачі інформації;
• передача інформації по каналу;
• перевірка одержуваної інформації та виправлення помилок;
• відключення каналу при його несправності та відновлення передачі після ремонту;
• генерація сигналу закінчення передачі та переведення каналу в пасивний стан.

Так як на фізичному рівні пересилаються просто сигнали, то при цьому не враховується, що в деяких мережах, в яких лінії зв'язку використовуються поперемінно кількома парами взаємодіючих комп'ютерів, фізичне середовище передачі може бути зайняте. Тому одним із завдань канального рівня є перевірка доступності середовища передачі. Іншим його завданням є реалізація механізмів виявлення та корекції помилок. Для цього на канальному рівні біти

групуються в набори, звані кадрами. Канальний рівень забезпечує коректність передачі кожного кадру, вміщуючи спеціальну послідовність бітів в початок і кінець кожного кадру, щоб відзначити його, а також обчислює контрольну суму, підсумовуючи всі байти кадру певним способом і додаючи контрольну суму до кадру. Коли кадр приходить, одержувач знову обчислює контрольну суму отриманих даних і порівнює результат з контрольною сумою з кадру. Якщо вони збігаються, кадр вважається правильним і приймається. Якщо ж контрольні суми не збігаються, то фіксується помилка.

Канальний рівень призначений для виконання наступних вимог мережевого рівня:

Незалежність від використовуваного середовища передачі	Мережевий рівень звільняється від усіх проблем, пов'язаних з тим, якого типу і якості канали використовуються для передачі інформації, яка конфігурація встановлюємих з'єднаннь, а також які режими передачі по даному з'єднанню задіюються
Незалежність від коду переданих даних	Канальний рівень повинен надавати можливість передачі даних і керуючої інформації верхніх рівнів по з'єднанню незалежно від того, в якому первинному коді вони представлені
Надійний обмін даними	При використанні канального рівня ймовірність появи в переданих користувачем даних вставок, втрат і спотворень досить малі. Крім того, можливо і вимога збереження порядку проходження переданих по з'єднанню даних

Розрізняють три види протоколів канальних рівнів, представлених на рис. 6.

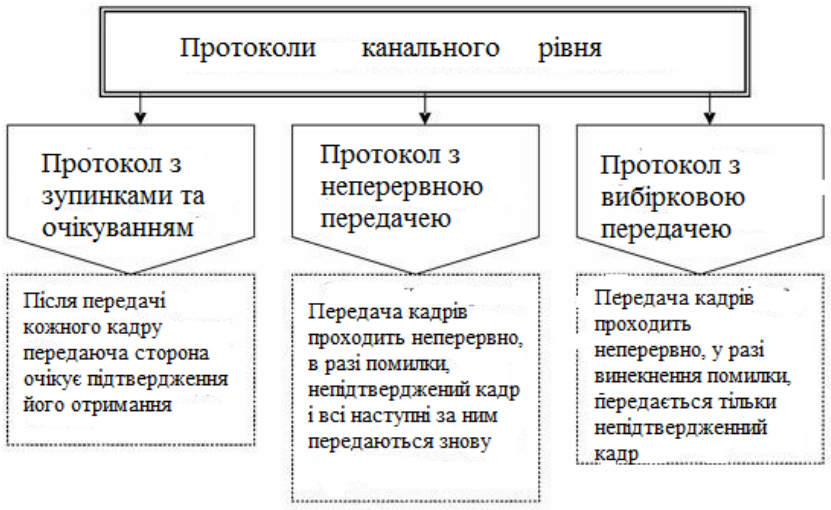

Рисунок 6 - Види протоколів канального рівня

Протокол з зупинками і очікуванням характеризується тим, що одночаоно може передаватися тільки один кадр, після чого приймаюча сторона чекає підтвердження. Якщо надійде негативне підтвердження або станеться прострочення часу очікування відповіді, кадр передається повторно. Пакет

скидається з накопичувача сторони, яка передає, лише після отримання позитивного підтвердження. Цей протокол підходить для напівдуплексних каналів зв'язку.

При використанні протоколу з безперервною передачею кадри передаються безупинно без очікування підтвердження. При отриманні негативного підтвердження або закінчення встановленого часу очікування непідтверджений кадр і всі наступні кадри передаються знову. Цей протокол більш продуктивний і передбачає використання дуплексного зв'язку.

У разі використання протоколу з вибіркової передачею повторна передача потрібна тільки для кадру, про який надійшло негативне підтвердження або для якого минув встановлений час очікування відповіді. Однак на приймаючій стороні потрібно накопичувач з переналаштуванням, так як в цьому випадку кадри можуть повторно передаватися і прийматися не по порядку. Через збільшення вартості реалізації протокол вибіркового повторення не знайшов широкого поширення.

У локальних мережах протоколи канального рівня використовуються комп'ютерами, мостами, комутаторами і маршрутизаторами. У комп'ютерах функції канального рівня реалізуються спільними зусиллями мережевих адаптерів і їх драйверів.

Мережевий рівень служить для утворення єдиної транспортної системи, що об'єднує декілька мереж, причому ці мережі можуть використовувати різні принципи передачі повідомлень між кінцевими вузлами і володіти довільною структурою зв'язків.

Призначення мережевого рівня полягає у встановленні, підтримці і роз'єднання мережевих з'єднань між об'єктами транспортного рівня і прокладання оптимальних маршрутів для передачі пакетів даних через топологію підмереж зв'язку.

Функції мережевого рівня:

• встановлення мережевого з'єднання для передачі даних об'єктів транспортного рівня в формі блоків даних;

• надання мережевих адрес, які використовуються для ідентифікації транспортних об'єктів;

• забезпечення незалежності передачі даних від використовуваних засобів передачі;

• попереднє узгодження параметрів якості обслуговування між користувачем-передавачем і постачальником мережевої служби.

• управління швидкістю передачі блоків даних мережевого рівня з боку приймача інформації;

• наскрізна передача, тобто доставка блоків даних мережевого рівня в абонентських системах, причому всі функції по вибору маршруту передачі і комутації (ретрансляції) здійснюються на мережевому рівні;

• передача послідовності окремих термінових (срочних) сблоків даних мережевого рівня. Ці термінові блоки даних мають обмежену довжину, і їх передача через точки доступу до служби здійснюється за правилами, відмінними від правил передачі нормальних даних;

• виявлення та виправлення помилок передачі даних;

• повторне встановлення з'єднання;

• роз'єднання мережевого з'єднання або користувачами мережевого рівня, або постачальниками мережевої служби.

Однією з важливих задач мережевого рівня є *маршрутизація.* Протоколи маршрутизації вибирають оптимальні маршрути через послідовність з'єднаних між собою підмереж.

Протоколи канального рівня локальних мереж забезпечують доставку даних між будь-якими вузлами тільки в мережі з відповідною типовою топологією, наприклад зіркоподібної, шинної і т. д. Це жорстке обмеження не дозволяє будувати мережі з розвиненою структурою, наприклад, мережі, що об'єднують кілька мереж підприємства в єдину мережу, або високонадійні мережі, в яких існують надлишкові зв'язки між вузлами. Можна було б ускладнювати протоколи канального рівня для підтримки петлевидних надлишкових зв'язків, але принцип поділу обов'язків між рівнями призводить до іншого рішення. Щоб, з одного боку, зберегти простоту процедур передачі даних для типових топологій, а з іншого - допустити використання довільних топологій, вводиться додатковий мережевий рівень.

На мережевому рівні сам термін "мережа" наділяють специфічним значенням. В даному випадку під "мережею" розуміється сукупність комп'ютерів, з'єднаних між собою відповідно до однієї зі стандартних типових топологій, що використовують для передачі даних один з протоколів канального рівня, визначений для цієї топології.

Усередині мережі доставка даних забезпечується відповідним канальним рівнем, а ось доставкою даних між мережами займається мережевий рівень, який і підтримує можливість правильного вибору маршруту передачі повідомлення навіть в тому випадку, коли структура зв'язків між складовими мережами має характер, відмінний від прийнятого в протоколах канального рівня.

Мережі з'єднуються між собою спеціальними пристроями, званими *маршрутизаторами.* Щоб передати повідомлення від відправника, що знаходиться в одній мережі, одержувачу, що знаходиться в іншій мережі, потрібно здійснити деяку кількість транзитних передач між мережами, щоразу обираючи відповідний маршрут. Проблема вибору найкращого шляху називається маршрутизацією, і її рішення є однією з головних задач мережевого рівня. Ця проблема ускладнюється тим, що найкоротший шлях не завжди найкращий. Часто критерієм при виборі маршруту є час передачі даних по цьому маршруту; воно залежить від пропускної здатності каналів зв'язку і інтенсивності трафіку, яка може змінюватися з плином часу. Деякі алгоритми маршрутизації намагаються пристосуватися до зміни навантаження, в той час як інші приймають рішення на основі середніх показників за тривалий час. Вибір маршруту може здійснюватися і за іншими критеріями, наприклад надійності передачі.

Мережевий рівень вирішує також задачі узгодження різних технологій, спрощення адресації у великих мережах і створення надійних і гнучких бар'єрів на шляху небажаного трафіку між мережами.

На мережевому рівні визначаються два види протоколів:

- **Мережеві протоколи** реалізують просування пакетів через мережу

- **Протоколи обміну маршрутною інформацією (протоколи маршрутизації)** - за допомогою них маршрутизатори збирають інформацію про топологію міжмережевих з'єднань

Саме ці протоколи звичайно мають на увазі, коли говорять про протоколи мережевого рівня.

Протоколи мережевого рівня реалізуються програмними модулями операційної системи, а також програмними і апаратними засобами маршрутизаторів.

Транспортний рівень забезпечує додаткам або верхнім рівням моделі - прикладному та сеансовому - передачу даних з тим ступенем надійності, яка їм потрібна.

Призначення транспортного рівня: забезпечення надійного, послідовного обміну даними між користувачами з використанням мережевого рівня і управління потоком даних, щоб гарантувати правильний прийом блоків даних.

Функції транспортного рівня:

• встановлення транспортного з'єднання між користувачами;
• відображення транспортної адреси на мережевий рівень;
• передача блоків даних без обмеження їх довжини і вмісту;
• розподіл довгих повідомлень, що надходять від верхніх рівнів, на пакети даних;
• управління темпом обміну;
• формування первинних повідомлень з набору пакетів, отриманих через нижні рівні;
• визначення якості сервісу, яке потрібно забезпечити за допомогою мережевого рівня, включаючи виявлення та усунення помилок;
• роз'єднання транспортного з'єднання.

Транспортний рівень оптимізує використання наявної мережевої служби для забезпечення необхідних сеансовими об'єктами характеристик передачі з мінімальною вартістю. На шляху від відправника до одержувача пакети можуть бути спотворені або загублені. Хоча деякі додатки мають власні засоби обробки помилок, існують і такі, які вважають за краще відразу мати справу з надійним з'єднанням. Модель взаємодії відкритих систем визначає п'ять класів сервісу, наданих транспортним рівнем. Ці види сервісу відрізняються якістю наданих послуг:

• терміновістю;
• можливістю відновлення перерваного зв'язку;
• наявністю коштів об'єднання декількох з'єднань між різними прикладними протоколами через загальний транспортний протокол;
• здатністю до виявлення і виправлення помилок передачі, таких як спотворення, втрата і дублювання пакетів.

Вибір класу сервісу транспортного рівня визначається, з одного боку, тим, якою мірою завдання забезпечення надійності вирішується самими додатками і протоколами більш високих, ніж транспортний, рівнів, а з іншого боку, цей вибір залежить від того, наскільки надійною є система транспортування даних в мережі, забезпечується рівнями, розташованими нижче транспортного - мережевим, канальним і фізичним. Так, наприклад, якщо якість каналів передачі зв'язку є дуже високою і ймовірність виникнення

помилок, не виявлених протоколами більш низьких рівнів, невелика, то розумно скористатися одним з полегшених сервісів транспортного рівня, не обтяжених численними перевірками та іншими прийомами підвищення надійності. Якщо ж транспортні засоби нижніх рівнів спочатку дуже ненадійні, то доцільно звернутися до найбільш розвиненого сервісу транспортного рівня, який працює, використовуючи максимум засобів для виявлення і усунення помилок, - за допомогою попереднього встановлення логічного з'єднання, контролю доставки повідомлень по контрольних сумах і циклічній нумерації пакетів , встановлення тайм-аутів доставки і т. ін.

Функціонування транспортного рівня розбивається на три фази:

1. Фаза встановлення з'єднання.
2. Фаза передачі даних.
3. Фаза роз'єднання з'єднання.

У фазі встановлення з'єднання виконується:

• вибір мережевого з'єднання, найбільш задовольняє вимогам сеансового об'єкта з урахуванням вартості і якості обслуговування;
• рішення про доцільність об'єднання або розщеплення транспортного з’єднання з метою оптимізації використання мережевих з'єднань;
• вибір оптимального розміру транспортного блоку даних протоколу;
• вибір функцій, які будуть задіяні в фазі передачі даних;
• відображення транспортних адрес в мережеві.

У фазі передачі даних здійснюється доведення транспортних блоків даних служби до сеансових об'єктів-одержувачів по транспортному сполученню передачею транспортних блоків даних протоколу. При цьому можуть бути задіяні такі функції, використання кожної з яких узгоджується в фазі встановлення з'єднання:

• впорядкування;
• управління потоком;
• виявлення помилок;
• виправлення помилок;
• передача термінових даних;
• розмежування транспортних блоків даних служби;
• ідентифікація транспортних з’єднань.

У фазі роз'єднання з'єднання виконуються наступні функції:

• оповіщення про причини роз'єднання;
• ідентифікація роз'єднуємого транспортного з’єднання передачі даних.

Фази функціонування транспортного рівня і виконувані функції представлені на рис. 7

Всі протоколи, починаючи з транспортного рівня і вище, реалізуються програмними засобами кінцевих вузлів мережі - компонентами їх мережевих операційних систем.

Протоколи нижніх чотирьох рівнів узагальнено називають мережевим транспортом, або транспортною підсистемою, т. як вони повністю вирішують задачу транспортування повідомлень із заданим рівнем якості в складених мережах з довільною топологією і різними технологіями. Решта три верхніх рівні вирішують задачі надання прикладних сервісів на основі транспортної підсистеми.

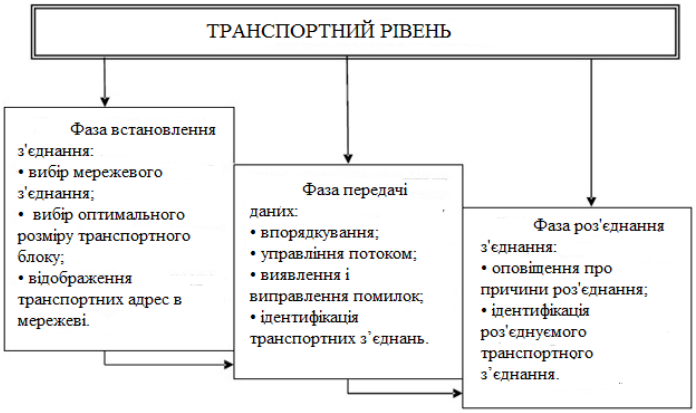

Рисунок 7 - Фази транспортного рівня

Сеансовий рівень встановлює, управляє і завершує сеанси взаємодії між прикладними завданнями.

Призначення сеансового рівня - забезпечувати управління діалогом для того, щоб фіксувати, яка зі сторін є активною в даний момент, а також надавати засоби синхронізації.

Функції сеансового рівня:

• формування наскрізного каналу зв'язку між взаємодіючими прикладними програмами;
• встановлення та розірвання сеансових з'єднань;
• вибір режиму передачі між прикладними процесами;
• управління черговістю передачі даних і їх пріоритетом;
• обмін нормальними даними;
• обмін терміновими даними;
• неподільна служба;
• управління взаємодією (маркери);
• синхронізація сеансу (контрольні точки);
• відновлення сеансу.

Одне із завдань сеансового рівня - перетворення імен в мережеві адреси, так що прикладні програми можуть використовувати імена для зв'язку з пристроями.

Неподільна служба - послуга сеансового рівня, за допомогою якої сеансні блоки даних служби, які надаслані по сеансовому з'єднанню, не надаються представницькому об'єкту-одержувачу до тих пір, поки це явно не дозволено представницьким об'єктом-відправником.

Управління взаємодією - послуга сеансового рівня, що дозволяє взаємодіючим представницьким об'єктам явно управляти черговістю виконання деяких управляючих функцій.

Виділяють наступні види взаємодії об'єктів:

- Двостороння одночасна (дуплексна) взаємодія - режим взаємодії, при якому обидва взаємодіючих представницьких об'єкта мають право одночасно передавати і приймати дані.

- Двостороння почергова (напівдуплексна) взаємодія - режим взаємодії, при якому взаємодіючі представницькі об'єкти по черзі отримують право передавати дані.
- Одностороння (симплексна) взаємодія - Режим взаємодії, при якому один з представницьких об'єктів тільки передає дані, а інший - тільки приймає

Синхронізація сеансового з'єднання - послуга сеансового рівня, що дозволяє представницьким об'єктам визначати і ідентифікувати точки синхронізації, здійснювати повторну установку сеансового з'єднання в заздалегідь певний стан і погоджувати точку повторної синхронізації.

На сеансовому рівні забезпечуються засоби, необхідні для організації та синхронізації діалогу між взаємодіючими представницькими об'єктами і для управління інформаційним обміном між ними. Для цього на сеансовому рівні встановлюються сеансове з'єднання між двома представницькими об'єктами і підтримується взаємодія з обміну даними. Сеансові з'єднання встановлюються за запитом представницького об'єкта, переданому в сеансові точки доступу до служби, і роз'єднуються або представницькими, або сеансовими об'єктами. У встановленому сеансовому з'єднанні підтримується діалог між представницькими об'єктами навіть при можливих втратах даних на транспортному рівні.

У кожен момент часу між сеансовими і транспортними з'єднаннями існує взаємно однозначна відповідність.

Передача термінових сеансових блоків даних служби зазвичай проводиться з використанням передачі термінових транспортних даних.

У разі виникнення відмов в транспортному сполученні сеансовий рівень може виконувати функції, необхідні для повторного встановлення транспортного сполучення з метою підтримки продовжує існувати сеансового з'єднання.

Сеансові об'єкти сповіщають (з використанням послуги сповіщення про особливі стани) представницькі об'єкти про те, що служба була перервана, і відновлюють службу тільки за вказівкою представницького об'єкта. Це дозволяє представницьким об'єктам провести повторну синхронізацію і продовжити функціонування з деякого узгодженого стану.

Роз'єднання сеансового з'єднання в нормальних умовах проводиться без втрат даних за запитом представницьких об'єктів. Сеансовий рівень також містить функції для передчасного роз'єднання сеансового з'єднання з можливими втратами даних.

Сеансові протоколи можуть здійснювати деякі функції з управління рівнем, такі, як активація і контроль помилок.

На практиці деякі додатки використовують сеансовий рівень, і він рідко реалізується у вигляді окремих протоколів, звичайно функції цього рівня об'єднують з функціями прикладного рівня і реалізують в одному протоколі.

Представницький рівень (рівень представлення даних) відповідає за те, щоб інформація, що посилається з прикладного рівня однієї системи, могла бути прочитаною прикладним рівнем іншої системи.

Призначення представницького рівня - забезпечення незалежності прикладних взаємодіючих об'єктів від використання конкретного синтаксису (кодування) інформації, що передається.

Функції представницького рівня:

• запит встановлення сеансу;
• вибір синтаксису;
• узгодження і повторне погодження синтаксису;
• перетворення синтаксису;
• передача даних;
• запит завершення сеансу.

Представницький рівень має справу з формою подання переданої по мережі інформації, не змінюючи при цьому її змісту. За рахунок рівня уявлення інформація, передана прикладним рівнем однієї системи, завжди зрозуміла прикладному рівню іншої системи. За допомогою засобів даного рівня протоколи прикладних рівнів можуть подолати синтаксичні відмінності в представленні даних або ж відмінності в кодах символів, наприклад кодів ASCII і EBCDIC. На цьому рівні може виконуватися шифрування і дешифрування даних, завдяки якому секретність обміну даними забезпечується відразу для всіх прикладних служб.

При необхідності, представницький рівень здійснює трансляцію між безліччю форматів представлення інформації шляхом використання загального формату представлення інформації.

На представницькому рівні забезпечується загальне уявлення даних, що використовуються між прикладними об'єктами. Таким чином забезпечується незалежність прикладних об'єктів від використовуваного синтаксису (тобто правил кодування переданої інформації). Синтаксична незалежність може бути досягнута двома способами:

1. Представницький рівень забезпечує загальні синтаксичні елементи, які використовуються прикладними об'єктами;

2. Прикладні об'єкти можуть використовувати будь-який синтаксис, а на представницькому рівні в цьому випадку здійснюється перетворення між різними формами синтаксису і загальним синтаксисом, необхідним для зв'язку між прикладними об'єктами. Це перетворення виконується у відкритій системі прозоро для інших відкритих систем і тому не впливає на стандартизацію протоколів представницького рівня.

Для задоволення вимог прикладних об'єктів на представницькому рівні може використовуватися будь-який синтаксис передачі, що підходить для цієї мети. Для досягнення інших цілей (наприклад, зменшення обсягу даних, що включає зниження вартості передачі) може проводитися перетворення синтаксису.

Прикладний рівень є кордоном між процесами мережі і прикладними (для користувача) процесами.

Призначення прикладного рівня - забезпечити доступ прикладних процесів до середовища передачі інформації для забезпечення їх взаємодії при вирішенні загальної задачі.

Функції прикладного рівня:

• ідентифікація партнерів, які передбачають взаємодіяти (наприклад, за допомогою імен, адрес, описів);

• визначення поточної готовності партнерів, які передбачають взаємодіяти;

• встановлення повноважень для передачі;

• узгодження механізму секретності;

• аутентифікація партнерів, які передбачають взаємодіяти;

• визначення методології призначення цін, достатності ресурсів, прийнятної якості обслуговування (наприклад, часу відповіді, відповідного рівня помилок);

• синхронізація взаємодіючих додатків;

• вибір дисципліни діалогу, що включає процедури ініціалізації і завершення;

• передача прикладних даних;

• узгодження відповідальності за виявлення помилок і процедур управління цілісністю даних;

• ідентифікація обмежень по синтаксису даних (безліч символів, структури даних).

Прикладний рівень - це найближчий до користувача рівень моделі взаємодії відкритих систем. Він відрізняється від інших рівнів тим, що не забезпечує послуг жодному з інших рівнів моделі; проте він забезпечує їм прикладні процеси, що лежать за межами масштабу моделі взаємодії відкритих систем. Прикладами таких прикладних процесів можуть служити програми обробки великомасштабних таблиць, програми обробки слів, програми банківських терміналів і т. ін.

Прикладний об'єкт складається з елемента користувача і елемента прикладної служби. Виділяються два типи елементів прикладної служби:

Загальні елементи - елементи, які надають можливості, необхідні безлічі додатків.

Спеціальні елементи - елементи, які надають можливості, необхідні для забезпечення додаткових послуг конкретним додатків (наприклад, передача файлів, банківські операції і т. ін.)

Прикладний рівень визначає мережеві прикладні програми, які обслуговують файли. Багато мережних програм-утилітів є частиною прикладного рівня.