Дизайн сети

По материалам лекций

Shivkumar Kalyanaraman

Информация, Компьютеры, Сети

- Информация: что-либо представленное в битах (*bits)*
  - *Форма* - может быть представлена битами, *и напротив*
  - *Вещество {Субстанция}* - не может быть представлена биты

- Свойства:
  - Бесконечно воспроизводимая
  - Компьютеры могут *"управлять"* информацией
  - Сети обеспечивают *"доступ"* к информации

Сети

- Потенциал работы с сетями:
  - Перемещение бит всюду, дешево и с желательными рабочими характеристиками
  - *Преодолевают пространственный барьер для информации*
- Сеть "обеспечивает связь" (соединения) (“connectivity”)

Что такое “Соединение” “*Connectivity*” ?

- Прямой или косвенный доступ к каждой другой вершине (узлу) в сети

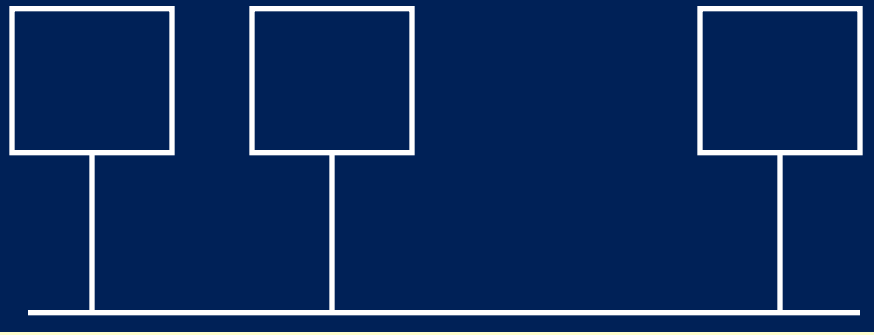
- Волшебство соединения проявляется, когда мы не имеем прямого «точка-точка» физического канала

  – *Чем жертвуем: рабочие характеристики хуже чем у истинного физического канала!*

Соединения → “Прямые”

- **Стандартные блоки**
  - **Связи (links): коаксиальный кабель, оптическое волокно …**
  - **Узлы: универсальные рабочие станции...**

- ***Прямое соединение:***
  - **point-to-point**
  - **multiple access**



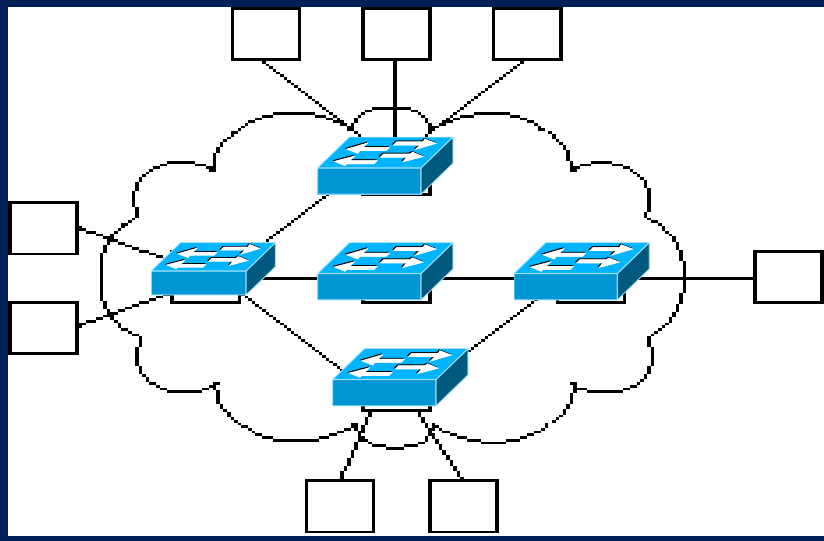
Соединения → “Косвенные”

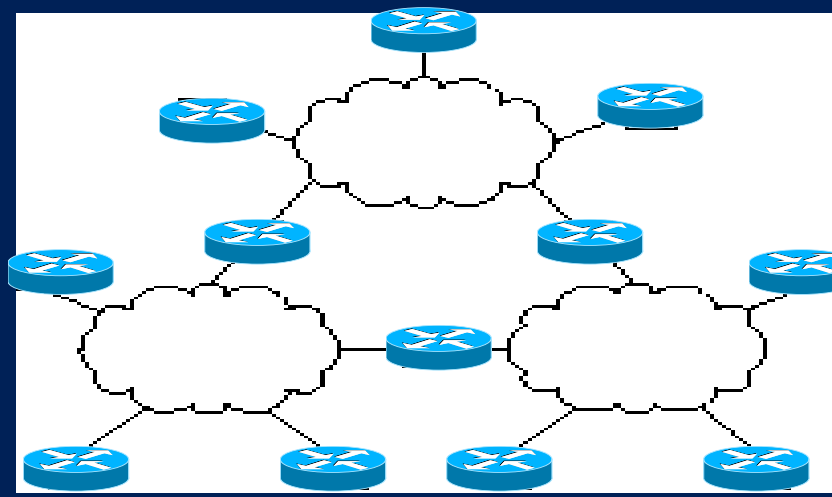
- *Косвенное соединение*
  - **коммутируемые сети**

=> Коммутаторы (*switches*)

  - **международные сети (inter-networks)**

=> *маршрутизаторы* (*routers*)

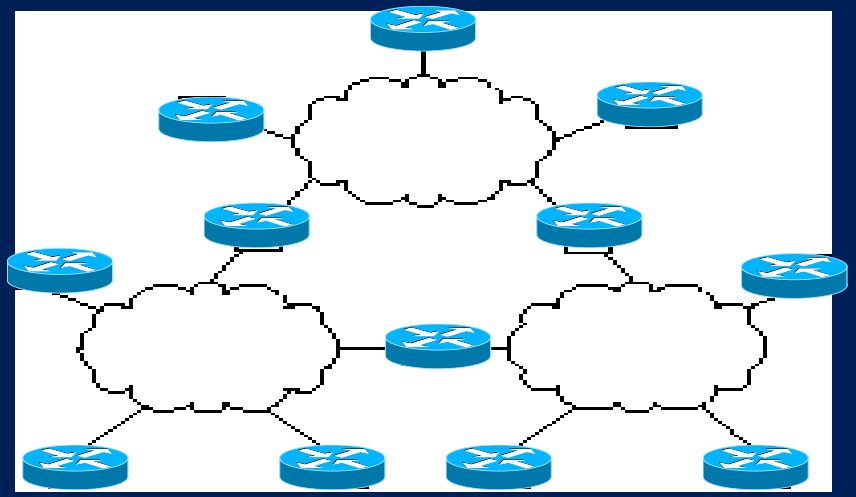




Соединения …

- **Internet:**
  - *Лучшее решение*

    (однако никаких гарантий характеристик)
  - *Пакет –за - пакетом*



- **Точка-точка физический канал :**
  - *Всегда связанный*
  - *Фиксирована полоса пропускания*
  - *Фиксирована задержка*
  - *Нулевой джиттер (Zero-jitter)*

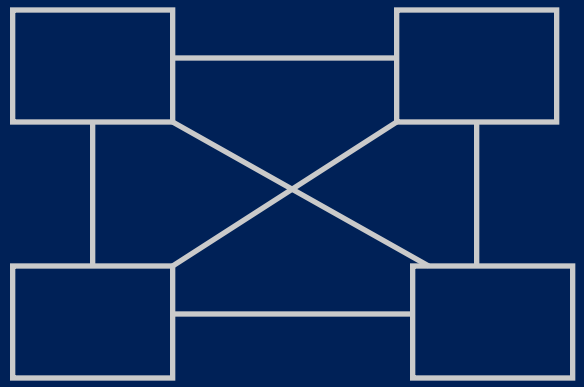
Point-to-Point Connectivity

- **Физический уровень: кодирование, модуляция и т.д.**
- **Канальный уровень необходим, если:**
  - **связь разделена между приложениями (кадрирование, управление доступом к среде передачи, мультиплексирование)**
  - **Связь ненадежна**
  - **связь используется спорадически, и трафик может затопить получателя (управление потоком данных)**
- **Никакой потребности в понятиях протоколах** подобно адресации, именам, маршрутизаторам, концентраторам, хабам, Forwarding (передача), фильтрации …

Соединение N пользователей: Прямо...

- ***Шина:*** *передача, столкновения, управление доступом к среде*
- ***Полносвязанная сеть****: Значительная стоимость против простоты использования*

Bus



Full mesh

- **Необходимо понятие адреса, если мы хотим чтобы каждый получал только ему предназначенные пакеты!**

Список проблем (пока)

- Топология
- Кадрирование
- Управление ошибками
- Управление потоком
- Множественный доступ
  - Как совместно использовать провод

Как строить Масштабируемые Сети?

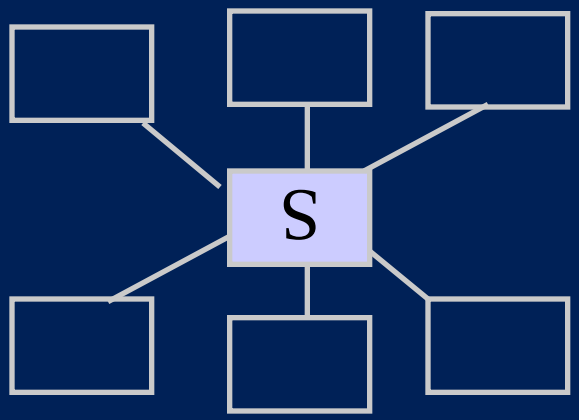
- Масштабирование: система должна позволять увеличивать ключевой параметр, увеличивать число хостов N …
  - *Неэффективность ограничивает масштабирование …*
- Прямые связи не эффективны и следовательно не масштабируемы
  - комментарий: неэффективны по причине необходимости огромного числа каналов связи
  - Примером решения этой проблемы является шинная архитектура: 1 дорогой общий канал, N дешевых подканалов
    - Неэффективны в использовании полосы пропускания

Filtering, forwarding …

- Фильтрация (filtering): выбор подмножества элементов из набора
  - *Фильтрация - ключ к эффективности и масштабированию*
- Передача (Forwarding): фактически передаваемые пакеты отфильтрованному подмножеству каналов/узлов
  - Пакет передается одному линку/узлу => эффективность
- Решение: Стройте узлы, которые фильтруют/передают (filter/forward) и соединяются косвенно => «коммутаторы» и «маршрутизаторы» (“switches” & “routers”)

Соединение N пользователей: косвенное (*Indirectly*)

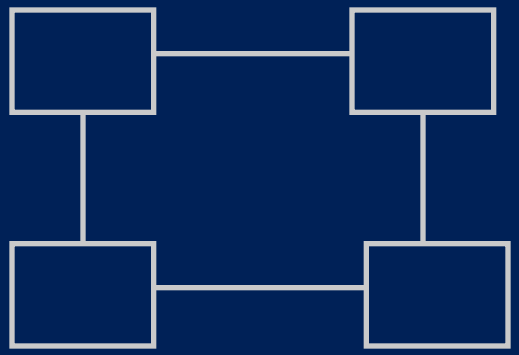
- Звезда : один hop path к любому узлу, надежность, функция передачи (forwarding)
- “Switch” S выполняет filter and forward!
  - Switch может параллельно предавать пакеты для повышения эффективности!



Star (звезда)

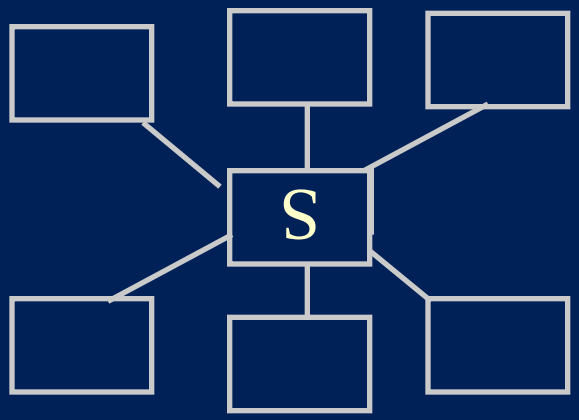
Соединение N пользователей: косвенное (*Indirectly*)

- Кольцо: надежность зависит от отказов в линии, почти минимальные связи
- Все узлы выполняют “forwarding” и “filtering”

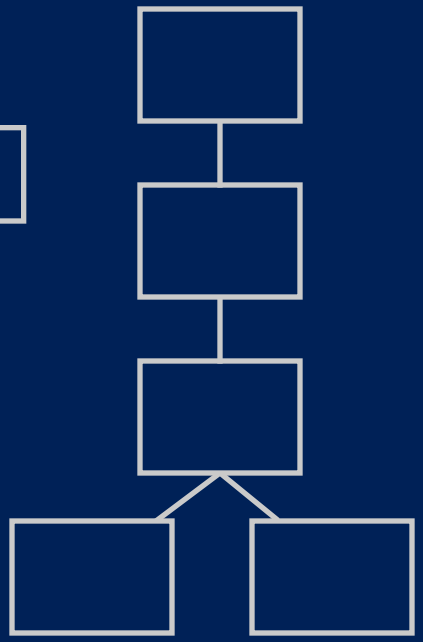


Ring

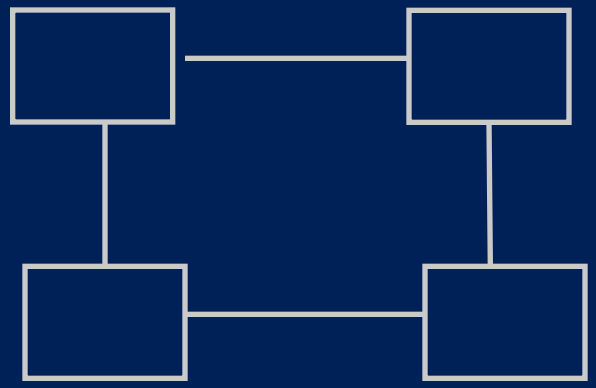
Топологии: косвенные соединения



Star

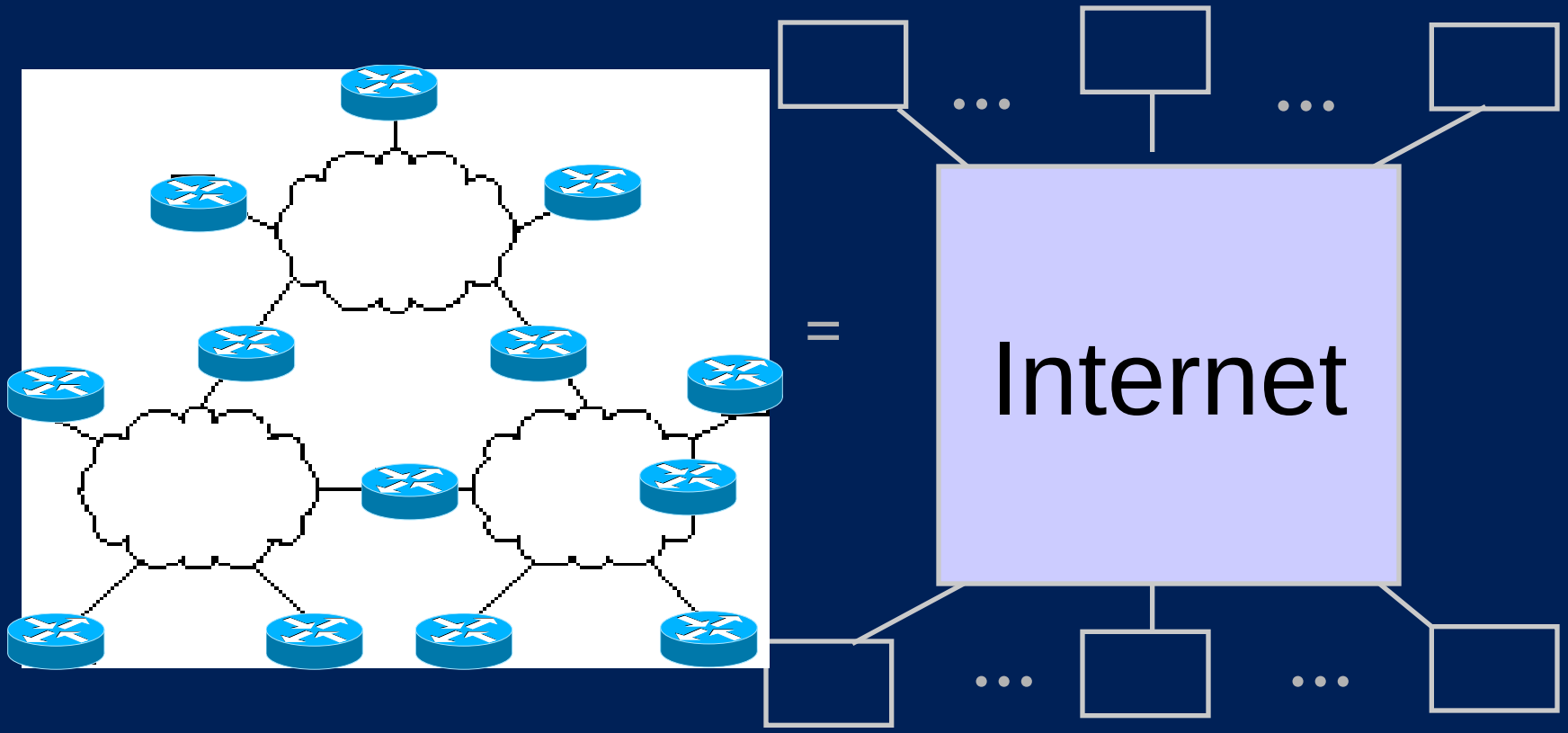


Tree



Ring

Inter-Networks: *Networks of Networks*



Наша цель – рассмотрение способов построения/проектирования " черный ящик " справа

Inter-Networks: Networks of Networks

- **Межсетевая организация вовлекает две фундаментальных проблемы:** *разнородность и масштабируемость*
- **Используемые концепции:**
  - **Трансляция, оверлей, разрешение адресов и имен, фрагментация: обрабатывать *разнородность***
  - **Иерархическая адресация, маршрутизация, именование, локализация адресов, управление перегрузкой: поддерживать *масштабирование***
- **Освещено более подробно в курсе «ККС»**

Дополнительные проблемы

- Фрагментация
- Коммутация, мост, маршрутизация
- Именование, адресация
- Управление перегрузкой, управление трафиком
- Надежность

Как выполняется проектирование таких систем?

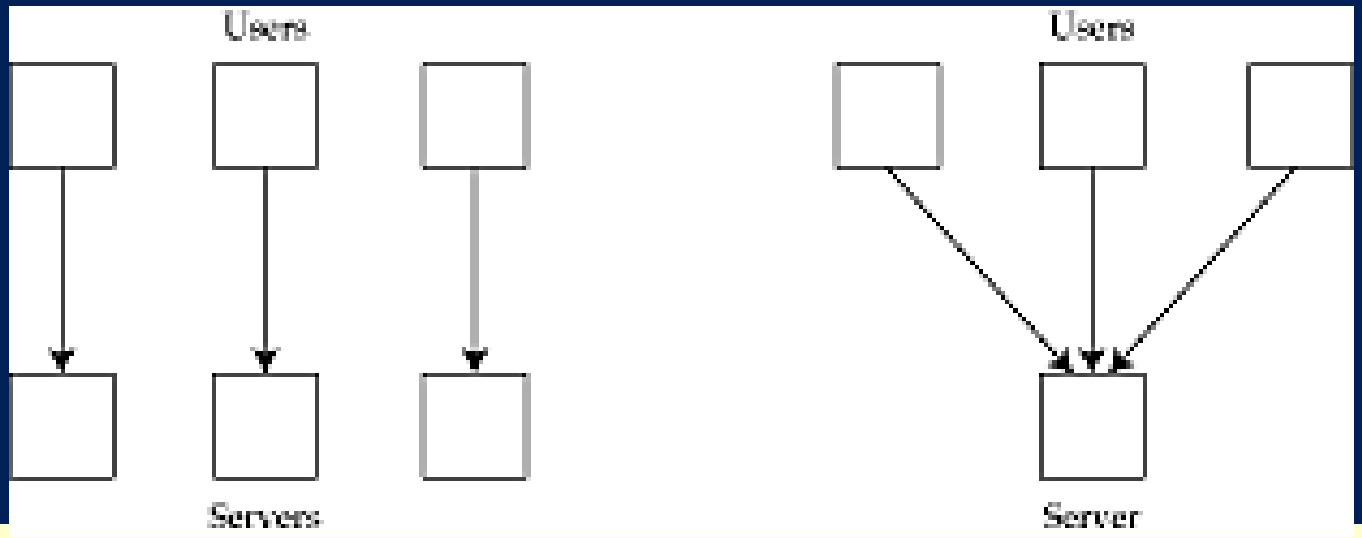
- Цель: Проектируем Международную сеть

- *Ресурсы* (Resources):
  - Пространство/память
  - Время
  - Вычисления
  - Деньги
  - Рабочая сила

- *Дизайн: использование более дешевых ресурсов против дорогих, чтобы достигнуть цели*

Стандартное решения: Мультиплексирование

- Мультиплексирование = совместное использование (sharing)
  - Продавать время и “пространство” для денег
  - *Цена вопроса*: время ожидания, буферное пространство/память и потери пакетов
  - *Выгода*: Деньги => в целом использование таких систем обходится для каждого меньшей суммой

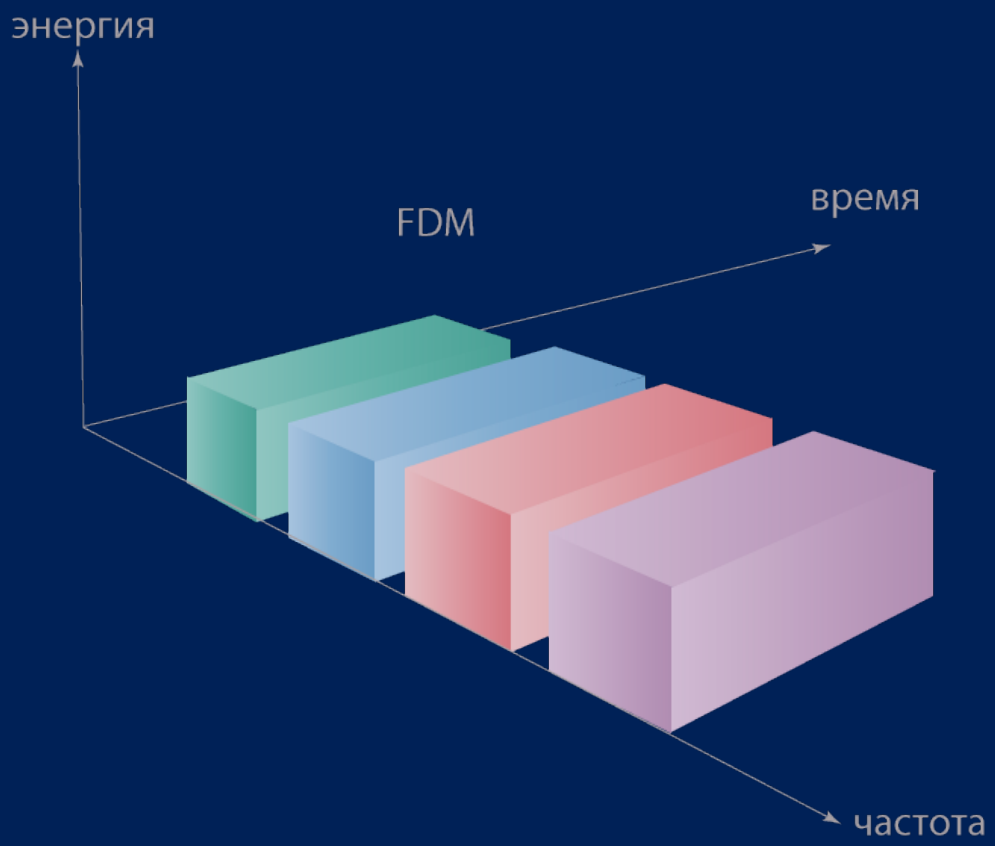


Мультиплексирование

- **Ресурс среды передачи: время, пространство, частота или код**
- **Необходимо передать несколько потоков данных для одного или нескольких пользователей**
  - Необходимо решить задачу множественного доступа к середе
  - Необходимо уплотнить потоки или спроектировать такой алгоритм, чтобы лучшим образом использовать ресурс среды передачи в имеющихся условиях
  - В литературе эту проблему именуют и как мультиплексирование, или как уплотнение, или как множественный доступ (MAC)
- **Пространственное разделение потоков - относительно простое решение задачи. Примеры:**
  - технология MIMO (англ. Multiple Input Multiple Output). Суть заключается в использование нескольких антенн, которые разносят друг от друга, чтобы они не мешали передачи.
  - Также наиболее простым примером пространственного разделения может служить ограничение мощности передатчиков или адаптивное изменение диаграмм направленности антенн

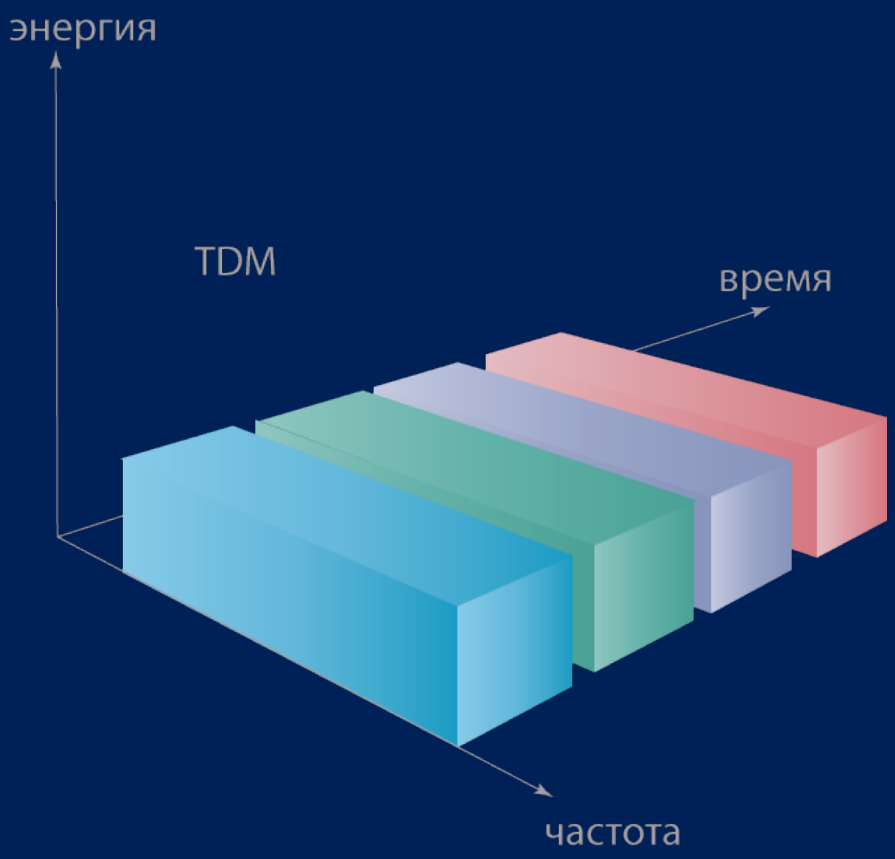
Частотное уплотнение
FDM (Frequency Division Multiplexing)

FDM (Frequency Division Multiplexing)



Временное уплотнение
TDM (Time Division Multiplexing)

TDM ((**Time** Division Multiplexing)



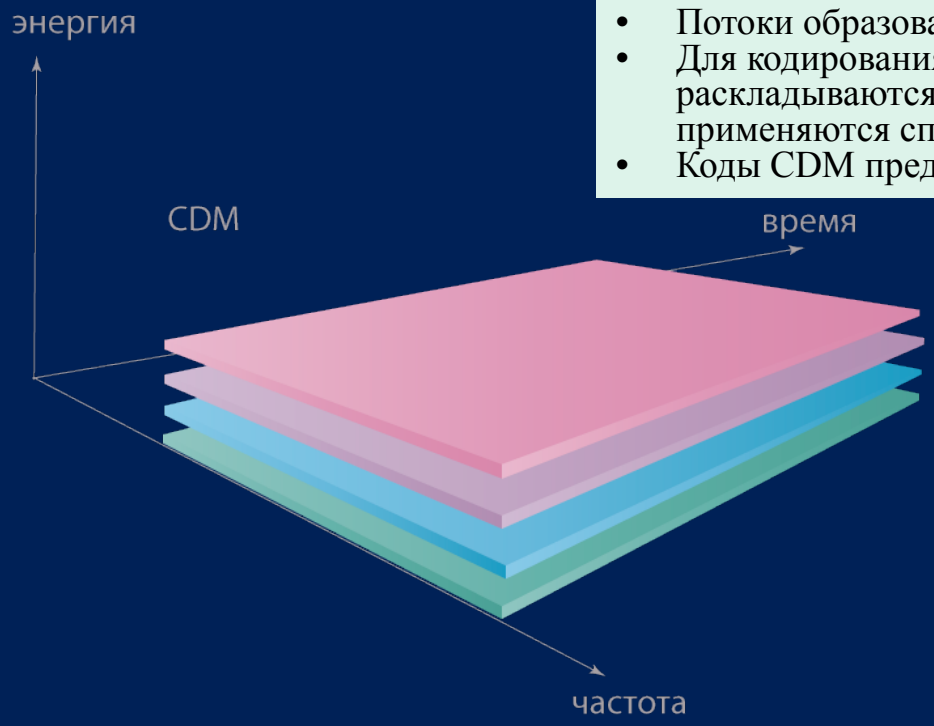
Кодовое мультиплексирование CDM (Code Division Multiplexing)

В данной схеме все передатчики транслируют сигналы на одной и той же частоте f, в области s и во время t, но с разными кодами c_i.

В схеме CDM каждый передатчик заменяет каждый бит исходного потока данных на CDM-символ - кодовую последовательность длиной в 11, 16, 32, 64 и т. п. бит (их называют чипами).

Кодовая последовательность уникальна для каждого передатчика. Как правило, если для замены "1" в исходном потоке данных используют некий CDM-код, то для замены "0" применяют тот же код, но инвертированный.

Приемник знает CDM-код передатчика, постоянно принимает все сигналы и оцифровывает их. Затем в специальном устройстве (корреляторе) производится операция свертки (умножения с накоплением) входного оцифрованного сигнала с известным ему CDM-кодом и его инверсией.. Если сигнал на выходе коррелятора превышает некий установленный пороговый уровень, приемник считает, что принял 1 или 0.



- Потоки образованы в одном частотно-временном интервале
- Для кодирования каждого потока сигналы, на которые раскладываются символы первоначальной последовательности применяются специальные коды.
- Коды CDM представляют собой ортогональные

Существуют различные модификации методики CDM, например:

- смесь CDM и FMD дают FHSS (Frequency Hopping Spread Spectrum)
- смесь CDM и TDM технику THSS (Time Hopping Spread Spectrum)
- FHSS, применяется
  - в Bluetooth,
  - в методе OFDM (Orthogonal frequency-division multiplexing.) - механизм мультиплексирования (уплотнения) посредством ортогональных поднесущих

Статистическое мультиплексирование

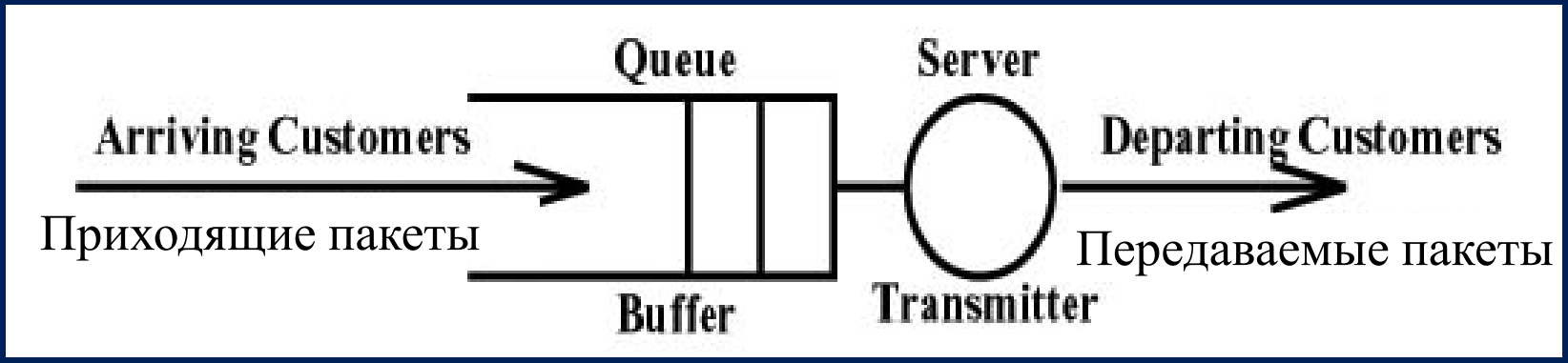
- Можно понизить требования к ресурсу, зная статистическую информацию о системе (сервисная скорость) и нагрузку на систему
  - Например: “средняя скорость <= сервисная скорость <= пиковая скорость”
  - Если “сервисная скорость < средняя скорость”, то система становится неустойчивой!!
- Поэтому, сначала дизайн (проект), чтобы гарантировать системную стабильность!!!
- Тогда, для устойчивой мультиплексной системы:
  - *Цель:*

    нагрузка = пик скорости / сервисная скорость = <1 (желательно меньше 1)
  - *Цена вопроса:*

Стоимость сервисной скорости = функция : буферизация, задержки в очередях, потери

Стабильность Мультиплексной Системы

Если	**→**	**средняя входная скорость > средней выходной скорости**
то	**→**	**система нестабильна**



Как гарантировать стабильность?

1. Резервировать (предусматривать) достаточные емкости, чтобы требований (запросов) было меньше чем зарезервированная емкость

2. Динамически обнаруживать перегрузку и приспосабливать или запросы или емкость буфера, чтобы избежать перегрузку

Каков компромисс производительности?

- Ситуация, в которой мы не можем получить кое-что ни для чего!
- Также известная как игра нулевой суммы

Если

- ✓ **R = полоса пропускания канала связи (bps - бит в сек)**
- ✓ **L = длина пакета (биты)**
- ✓ **a = средняя частота/скорость поступления пакетов**

То

- ✓ **Интенсивность трафика = La/R**

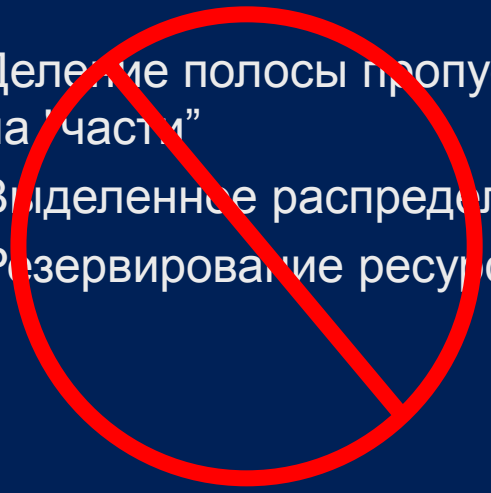
Выводы

- ✓ La/R ~ 0: малая средняя величина задержки в очереди
- ✓ La/R → 1: задержки становятся большими
- ✓ La/R > 1: бесконечность среднего времени ожидания (обслуживание ухудшается неограниченно => неустойчивость)!

Пример дизайна: коммутация пакетов

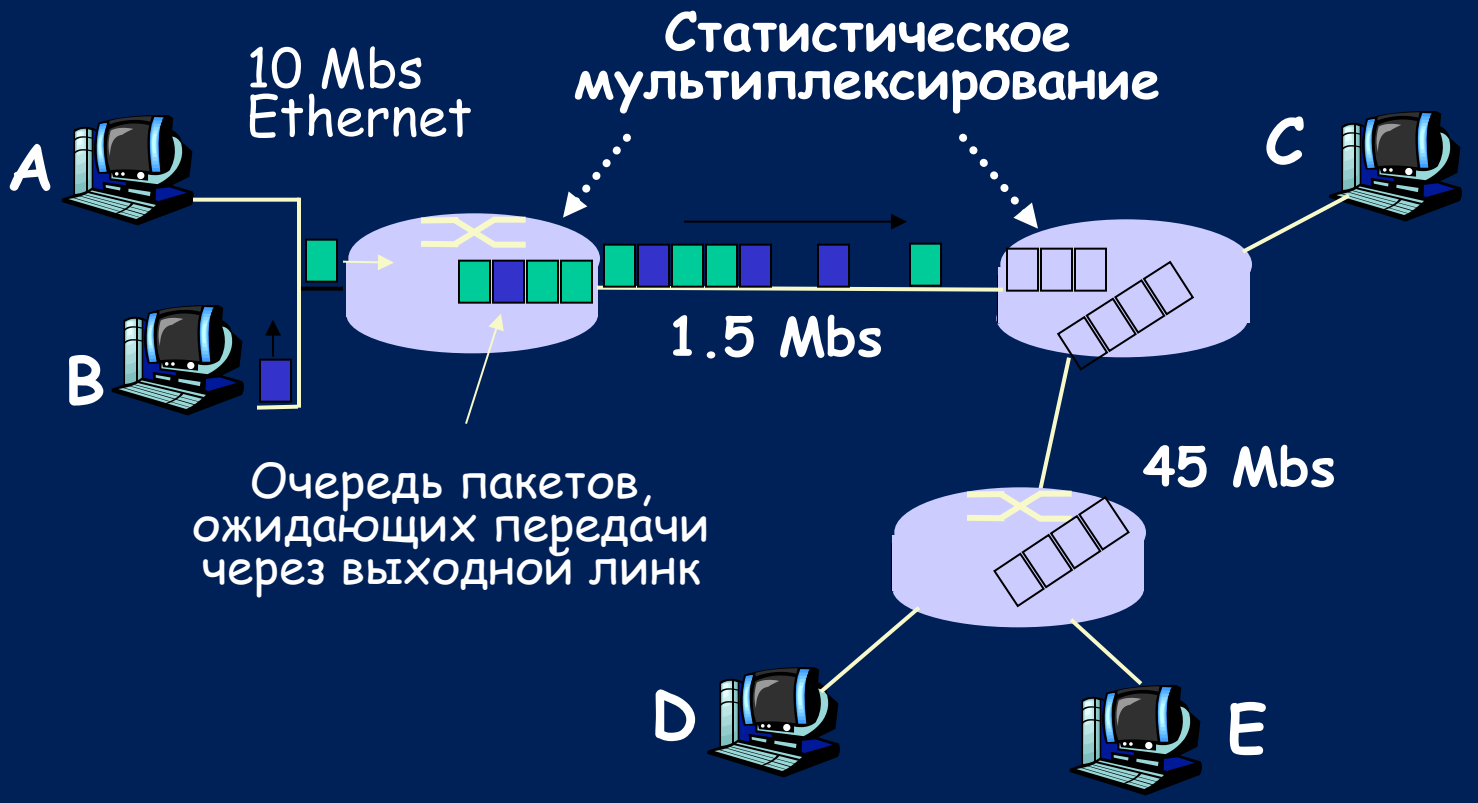
***Коммутация пакетов:* другая форма мультиплексирования**

- **Нарезаем данные (не полосу пропускания!) на "пакеты"**
  - **Пакеты = данные + метаданные (заголовок)**
- **Коммутируем пакеты в промежуточных узлах**
  - С промежуточным накоплением (Store-and-forward), если не располагаем в некоторый момент времени полосой пропускания



- Деление полосы пропускания на "части"
- Выделенное распределение
- Резервирование ресурса

Коммутация пакетов

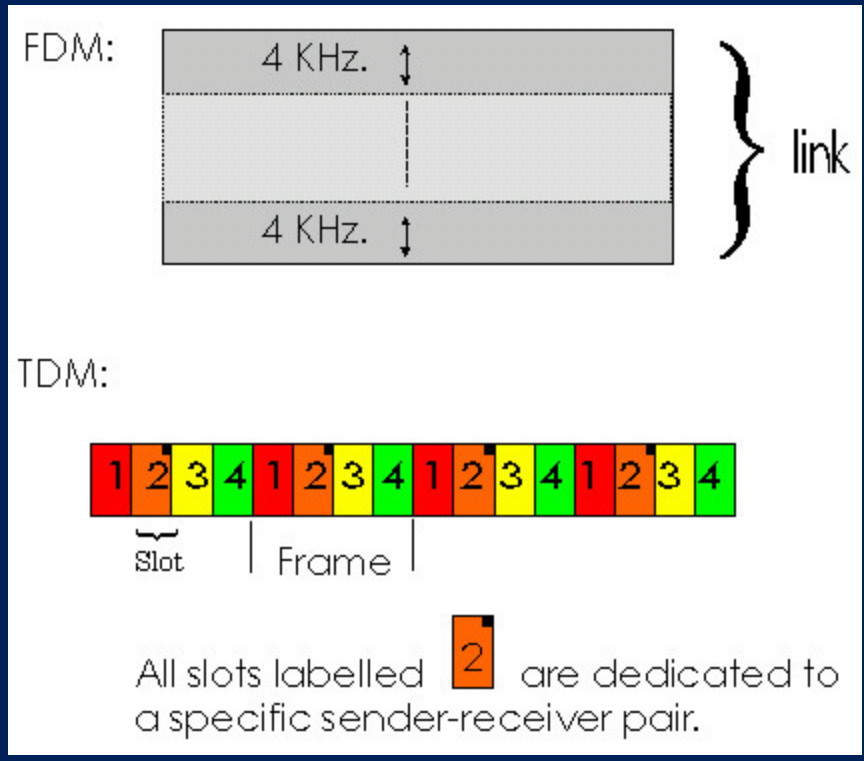


- Стоимость: каждый пакет самодокументирован заголовком, буферизация и задержки для приложений.
- необходимо или резервировать ресурсы или динамически обнаруживать/приспосабливаться, чтобы для стабильности избежать перегрузки

Пример дизайна: Коммутация каналов

Коммутация каналов: форма мультиплексирования

- **Делим полосу пропускания канала на "части"**
- Резервируем (предусматриваем) некоторую “часть” на поочередных каналах связи и связываем их вместе, чтобы формировать "схему” (цикл/кадр)
- Структура трафика определяется в зарезервированных каналах
- Ресурсы простаивают если не используются: дорого



— **Существование “схемы” (структуры цикла) позволяет обходиться без "заголовков"**

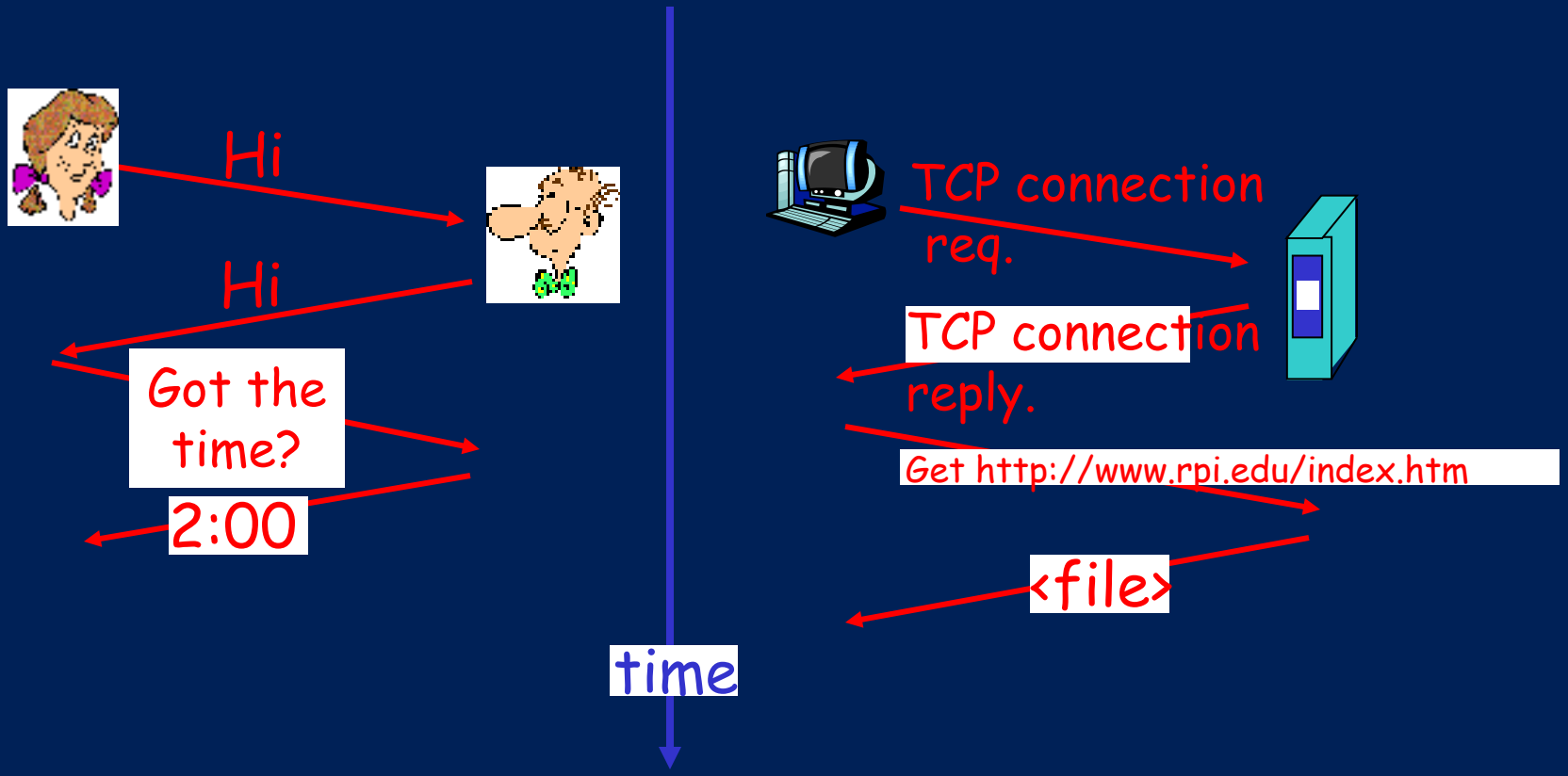
— **Все вышесказанное относится к вопросу «синхронизация»**

Резюме идей дизайна систем

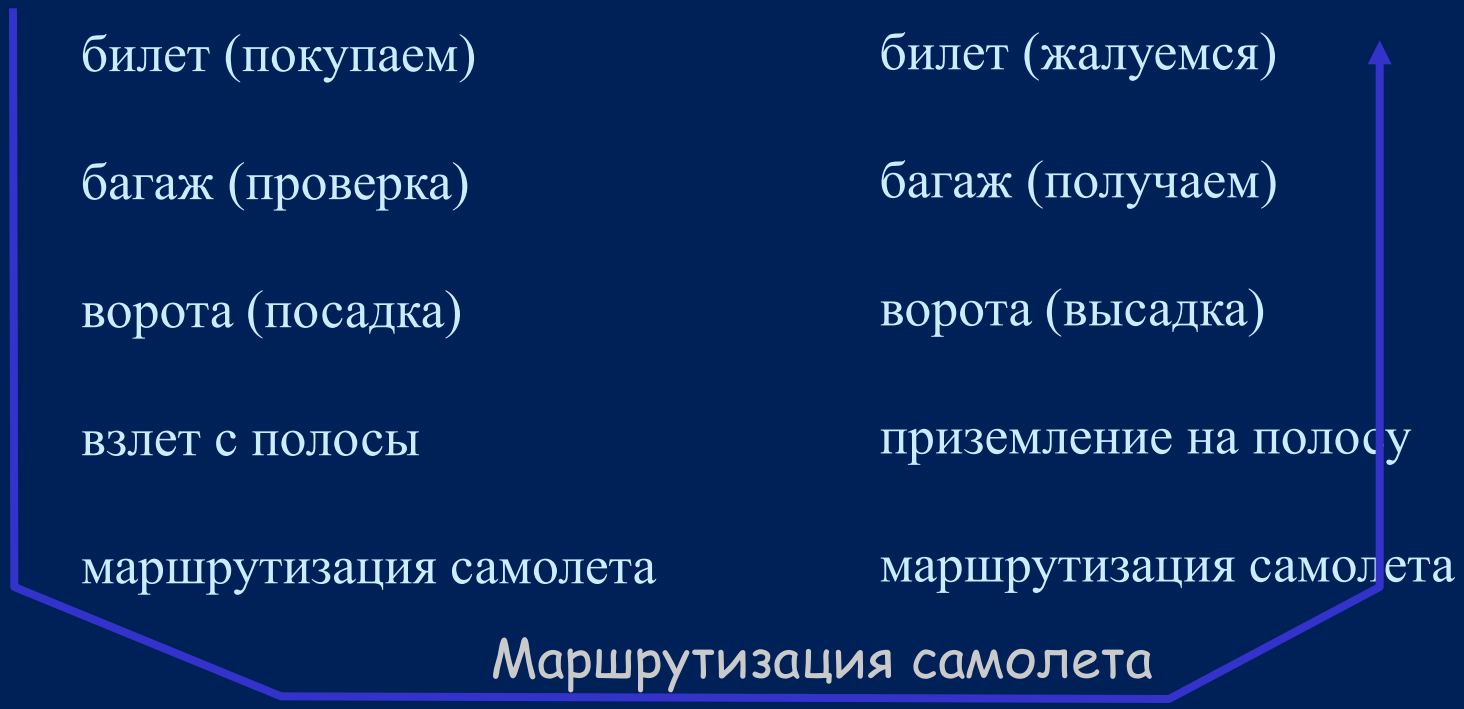
- Мультиплексирование
- Статистическое мультиплексирование
- Стабильность и качество работы (обмена)
- Коммутация каналов (Circuit switching)
- Пакетная коммутация (Packet switching)

Что такое протокол?

- Протоколы реализованы в сетевом программном обеспечении
- сравним: протокол человека и протокол сетевой:

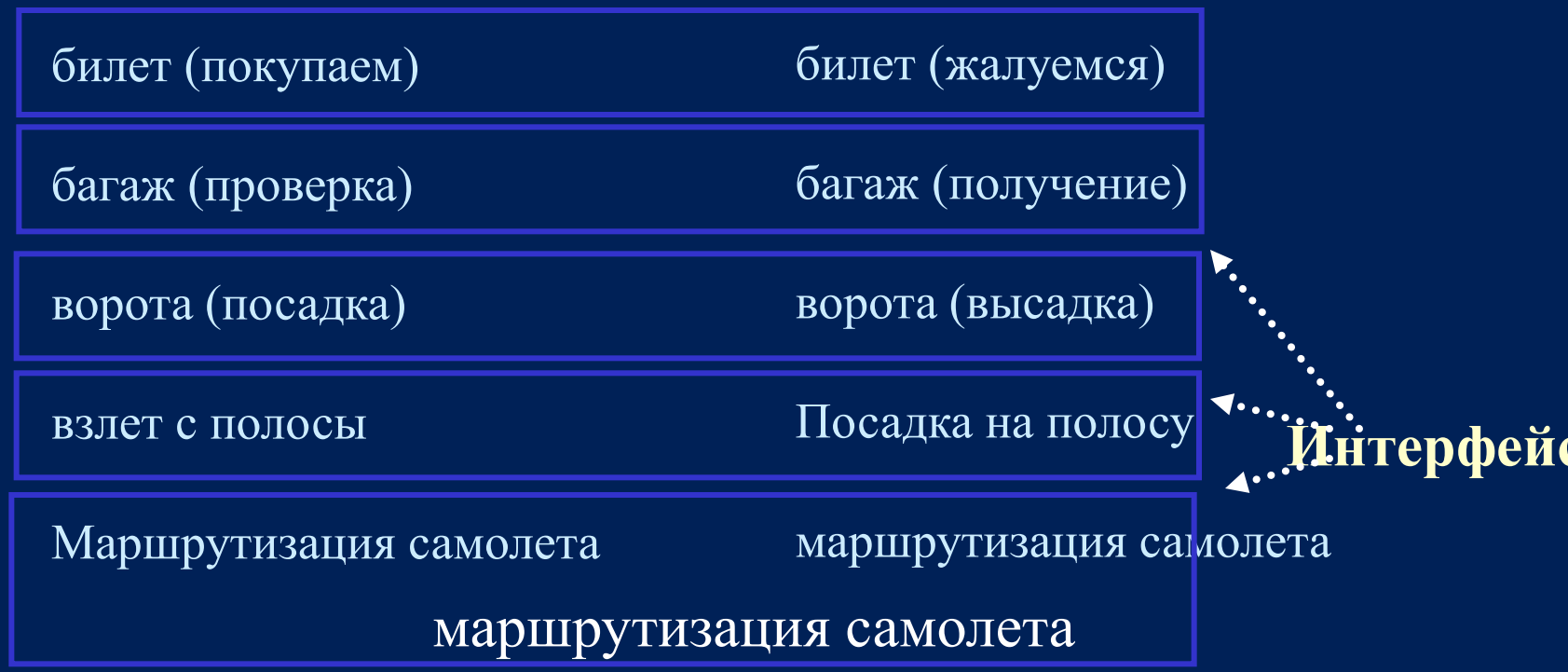


Аналогия: Организация путешествия самолётом



- Протоколы: последовательность функций, выполняемых в различных местах

Организация путешествия самолётом: раздельное представление



- *Уровни: каждый уровень предоставляет сервис*
  - через его собственные действия внутри уровня и
  - доверии к сервисам нижележащих уровней

Многоуровневое путешествие самолетом : сервисы (*services*)

Из страны в страну доставка персон + багажа

Прием и выдача багажа

Перемещение людей: посадка и высадка

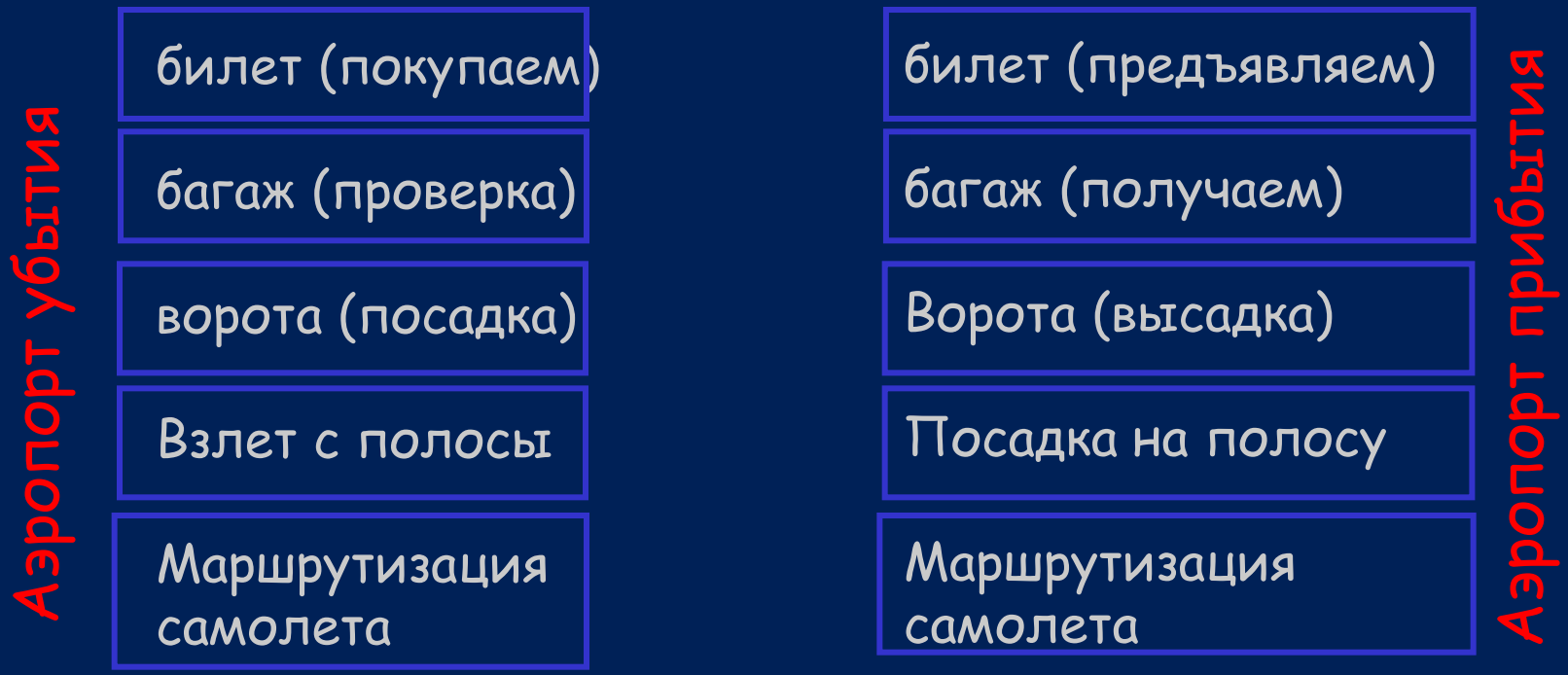
Предоставление взлетно-посадочной полосы

Маршрутизация самолета от источника к месту назначения (from source to destination)

Точно так же организованы сетевые протоколы

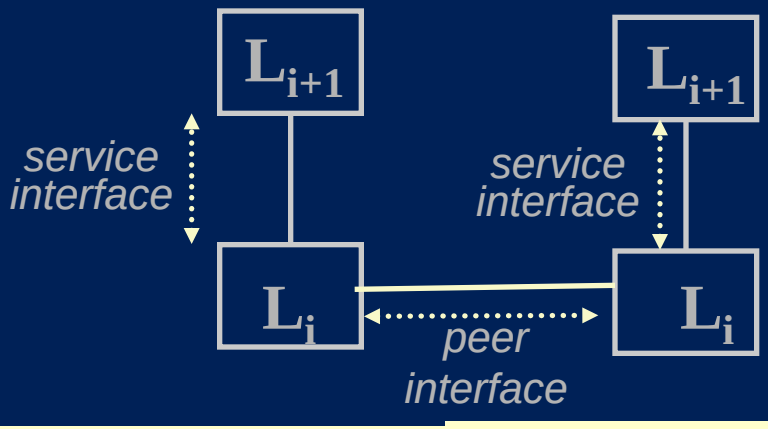
- **в связку уровней (стэк протоколов)!**

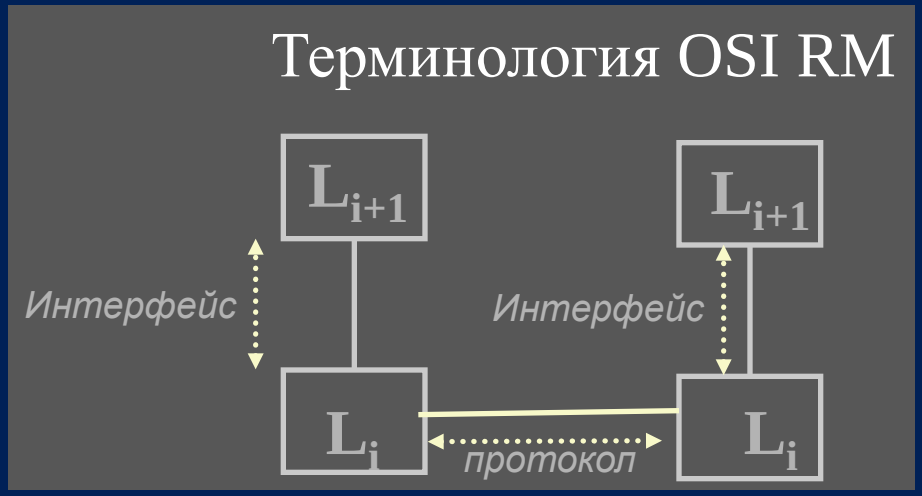
Многоуровневое распределенное выполнение



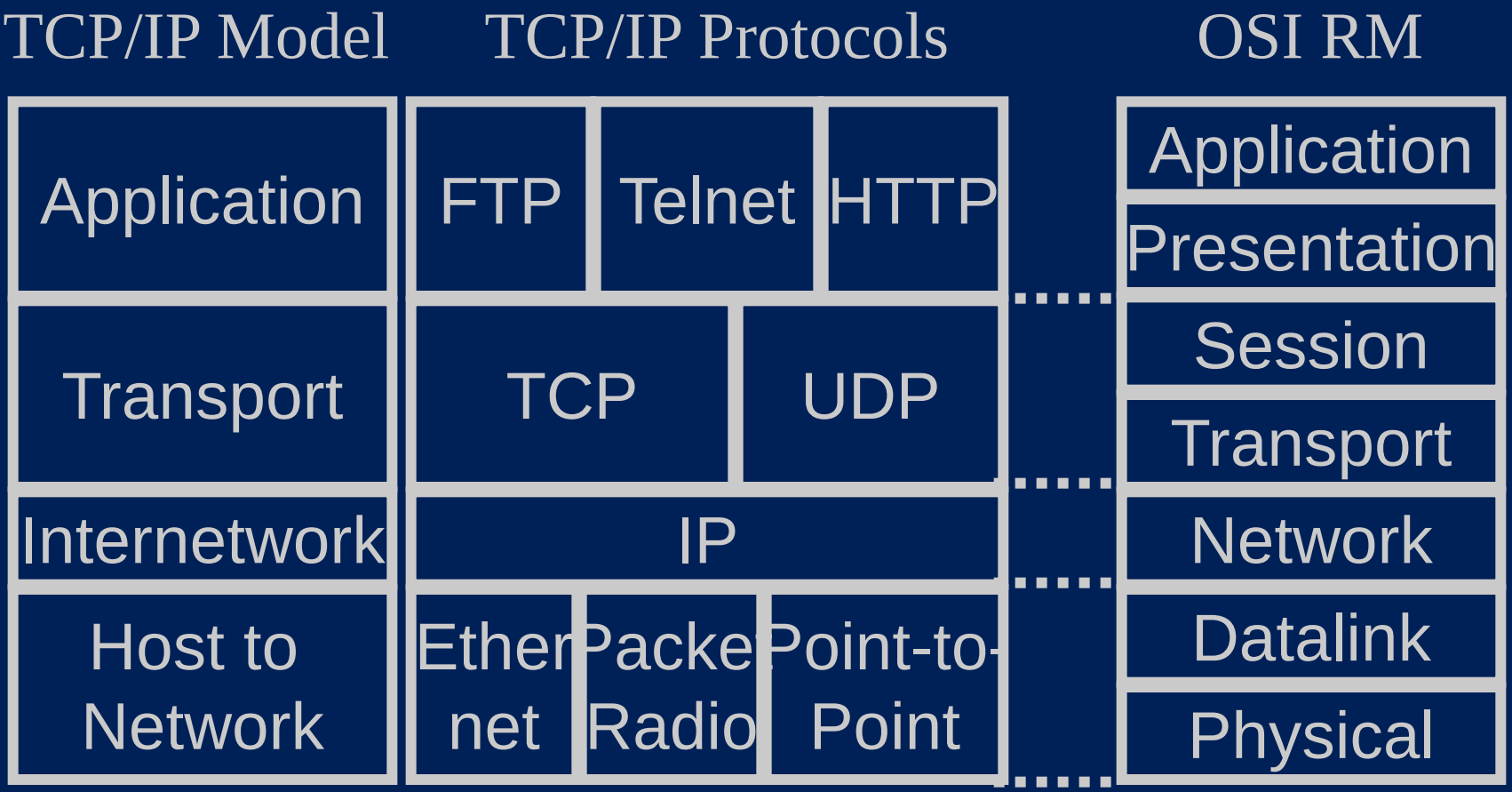
Действия (операции) протокола

- Существуют стандартные блоки сетевой архитектуры
- Каждый объект протокола имеет два различных интерфейса
  - *сервисный интерфейс (service interface)*: определяет операции протокола между смежными объектами в одной системе
  - *одноранговый интерфейс (peer-to-peer interface)*: определяет обмен сообщениями между перами (peer) [объектами одного уровня]





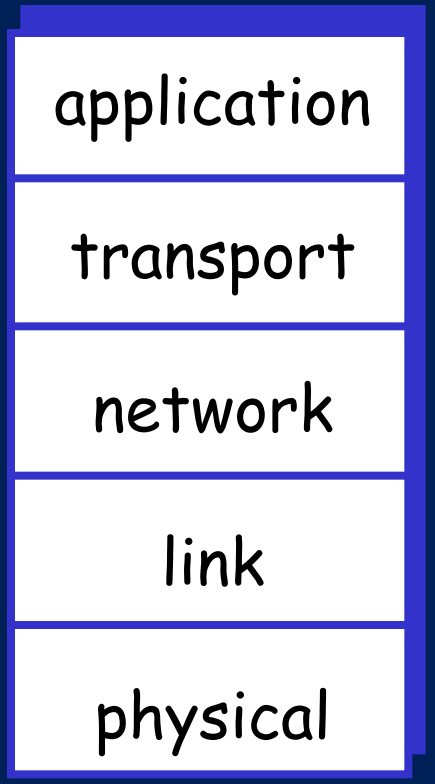
Многоуровневая эталонная модель (Reference Models-RM)



“Нисходящий” подход означает, что мы сначала изучим прикладной уровень и затем узнаем о нижних уровнях

Стек протоколов Internet

- **application: поддерживает сетевые приложения**
  - **ftp, smtp, http**
- **transport: host-host data transfer**
  - **tcp, udp**
- **network: routing of datagrams from source to destination**
  - **ip, routing protocols**
- **link: data transfer between neighboring network elements**
  - **ppp, ethernet**
- **physical: bits “on the wire”**

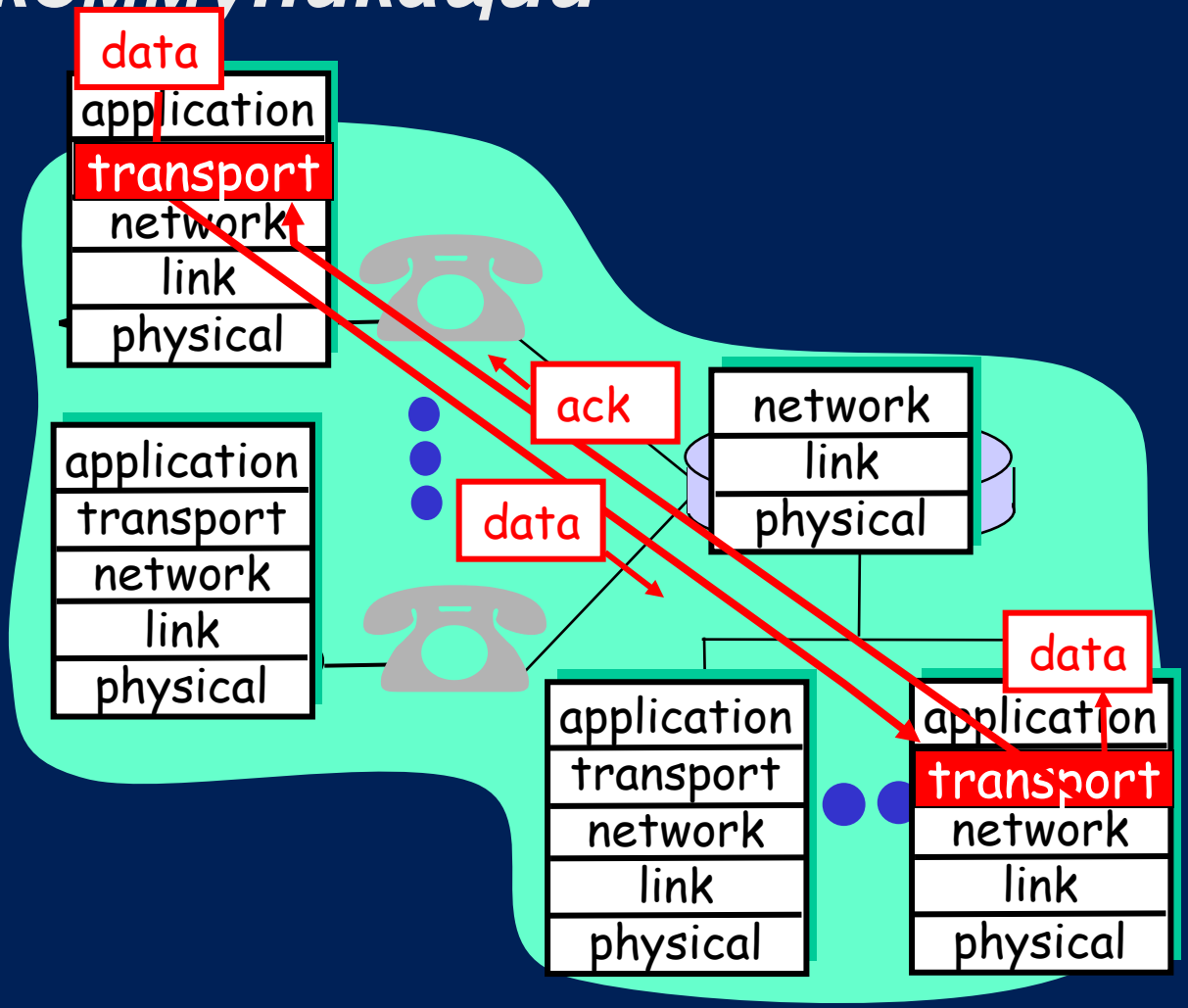


Многоуровневость: *логические коммуникации*

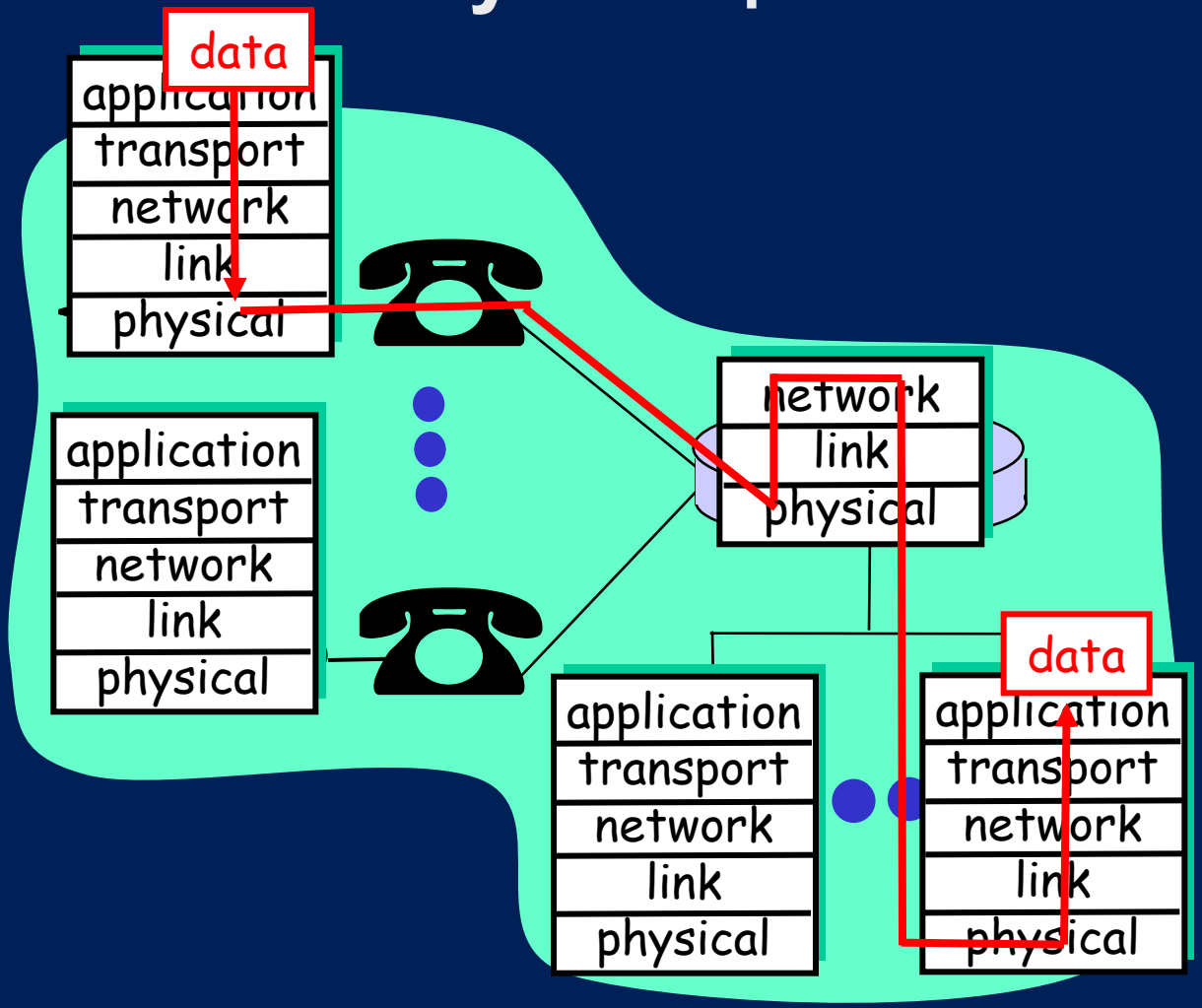
Например: transport

- **Поступают**
  - **данные от приложения**
- **Добавляются**
  - **адрес и проверочная последовательность для контроля правильности приема**
  - **Информация для формирования “датаграммы”**
- **Датаграмма посылается соседу**
- **Ожидаем от соседа положительного подтверждения**

- **Аналогия: почтовое уведомлением о получении**



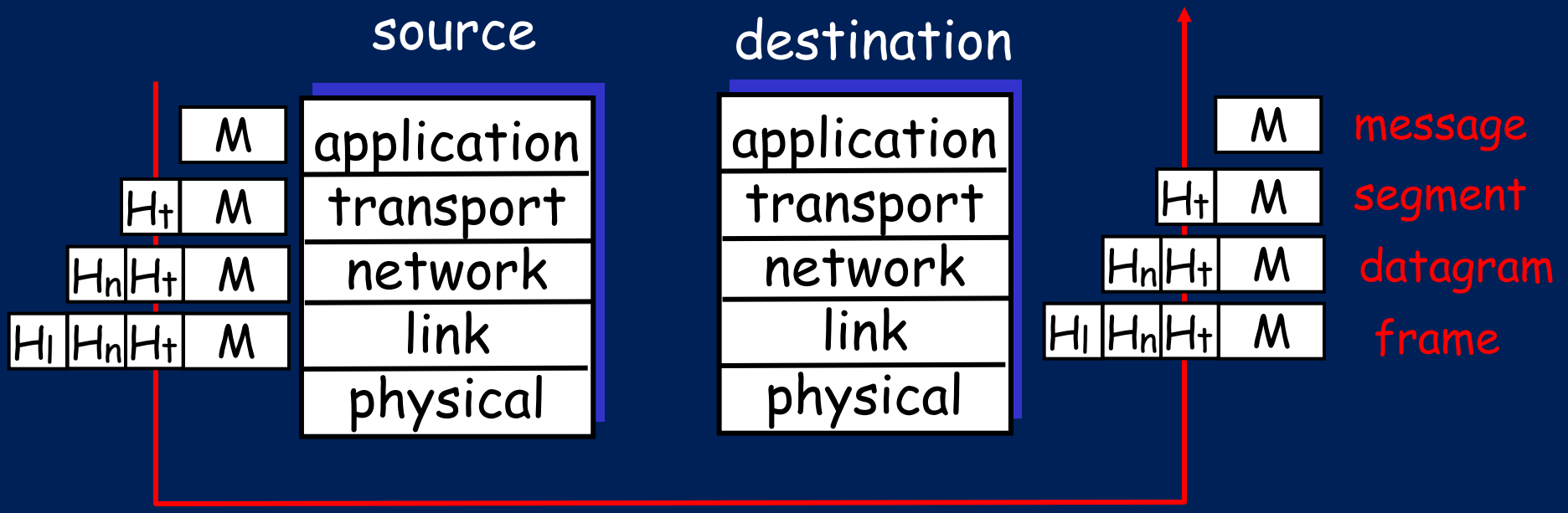
Многоуровневость: *физические* коммуникации



Многоуровневые протоколы и данные

Каждый уровень берет данные верхнего уровня

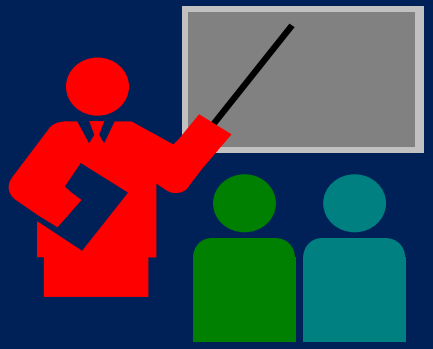
- **Добавляет к нему заголовок для создания нового блока данных (инкапсуляция -“encapsulation”)**
- **Передает сформированный новый блок данных нижележащему уровню**



Перспективы дизайна

- *Сетевые пользователи (Network users):* предоставляемый приложениям сервис должен гарантировать передачу каждого сообщения без ошибок в переделах определенного времени
- *Сетевые архитекторы (Network designers): рентабельный дизайн*, например, эффективное использование и справедливое распределение сетевых ресурсов между различным пользователями
- *Сетевые провайдеры (Network providers):* просто администрируемая и управляемая система , например, легкая изоляция неисправностей и ошибок разного рода

Резюме



- Администрирование, Администрирование
- Сети, коннективность, топологии ….
- Пучок сетевых концепций и проблем, которые исследуются в этом курсе ...

Источник

http://www.pde.rpi.edu/

Or

http://www.ecse.rpi.edu/Homepages/shivkuma/

Shivkumar Kalyanaraman

Rensselaer Polytechnic Institute

shivkuma@ecse.rpi.edu