Системное программное обеспечение локальных компьютерных сетей

Введение в сетевое ПО

Денис Пынькин

2013 – 2014

e-mail: denis.pynkin@bsuir.by

http://goo.gl/32cTB

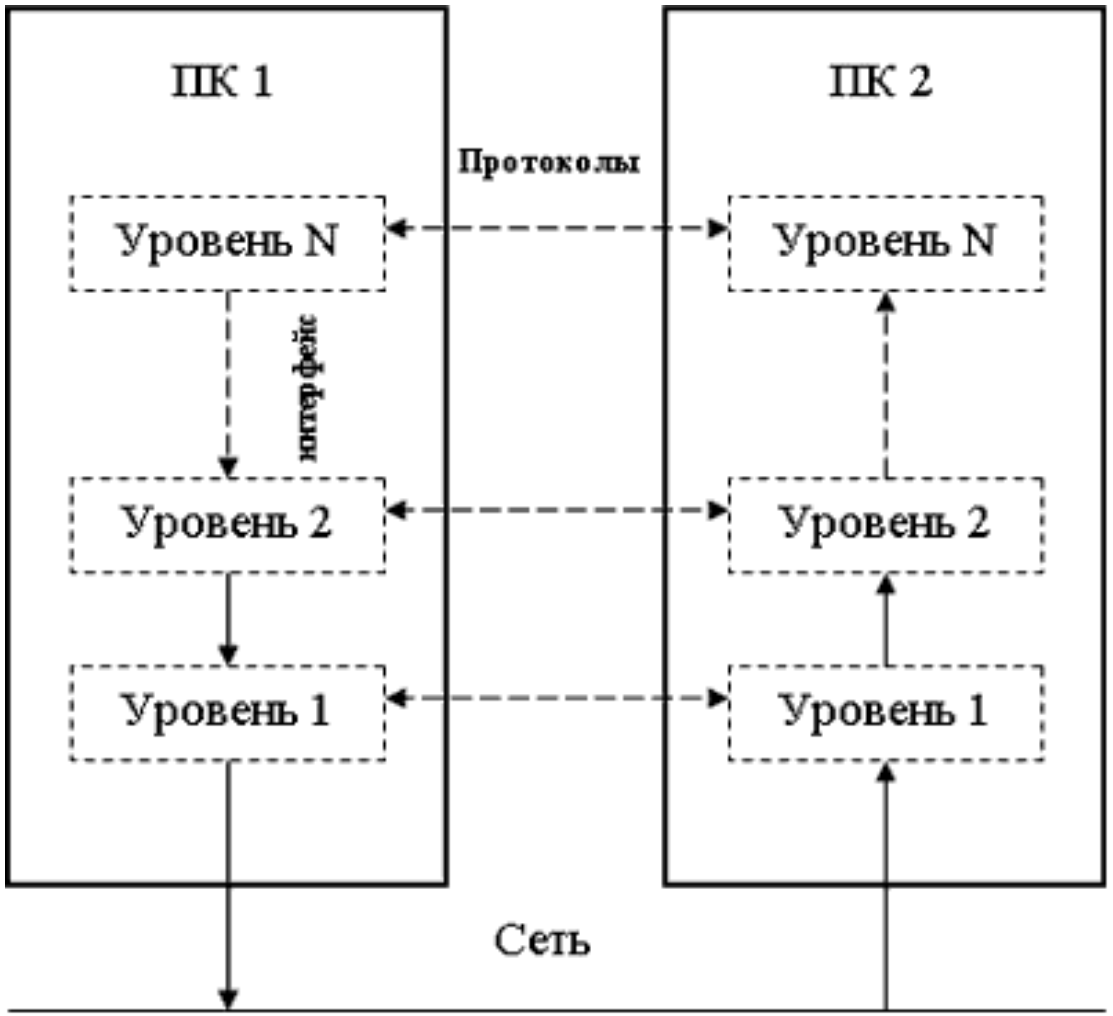
СЧАСТЬЕ ДЛЯ ВСЕХ, ДАРОМ, И ПУСТЬ НИКТО НЕ УЙДЕТ ОБИЖЕННЫЙ!

(c)Стругацкие, Пикник на обочине

Сетевое ПО

При создании первых ЛВС основное внимание уделялось аппаратуре, а вопросы программного обеспечения оставлялись на потом. Современное сетевое программное обеспечение в высокой степени структурировано.

Архитектура сети



Протокол

Уровень одной машины поддерживает связь с уровнем другой машины. Правила и соглашения, используемые в данном общении, называются **протоколом** уровня N.

С сетевой точки зрения В протокол включаются форматы обмена данными (кадры, сообщения) и правила обмена данными (последовательность)

Протоколы могут меняться, при условии, что предоставляемые ими сервисы останутся неизменными.

Список протоколов, используемых системой, по одному протоколу на уровень, называется **стеком протоколов**.

Интерфейс

Между парой смежных уровней находится **интерфейс**(служба), определяющий набор примитивных операций, предоставляемых нижним уровнем верхнему.

Когда разработчики решают, сколько уровней включать в сеть и что должен делать каждый уровень, одной из важнейших задач является определение ясных интерфейсов между ними. Эта задача требует, чтобы каждый уровень выполнял особый набор хорошо понятных функций. Служба описывает интерфейс между двумя уровнями, где нижний уровень является поставщиком сервиса, а верхний – потребителем.

Архитектура сети

Набор уровней и протоколов называется **архитектурой сети**. Спецификация архитектуры должна содержать достаточно информации для написания программного обеспечения и создания аппаратуры для каждого уровня, чтобы они выполняли требования протокола. Детали реализации и спецификации интерфейсов не являются частями архитектуры, поскольку они спрятаны внутри машины. При этом не требуется, чтобы интерфейсы были одинаковыми, главное, чтобы протоколы соответствовали спецификации.

Типы служб

Уровни могут предоставлять вышестоящим уровням услуги 2-х типов: с наличием или отсутствием соединения. При наличии соединения клиент сначала устанавливает соединение, а в конце разрывает его. При отсутствии соединения клиент просто отсылает свое сообщение, в котором содержится полный адрес получателя, при этом каждое сообщение может идти своим путем и порядок получения данных не гарантируется.

Службы ориентированные на соединение

- Надежный поток сообщений
- Надежный поток байт
- Ненадежное соединение

Службы без установки соединения

- Ненадежная дейтаграмма
- Дейтаграмма с подтверждением
- Запрос-Ответ

Примитивы служб

Служба формально описывается набором примитивов или операций, доступных пользователю или другой сущности для получения сервиса. Эти примитивы заставляют службу выполнять некоторые действия или служат ответами на действия сущности такого же уровня. Если набор протоколов входит в состав ОС, то примитивы являются системными вызовами.

Примитивы служб

Набор примитивов может быть разным и зависит от типа предоставляемого сервиса. Например, минимальный набор примитивов для простой передачи с установлением соединения будет выглядеть так:

- LISTEN (ожидание) – ожидание входящего соединения
- CONNECT (соединение) – установка соединенияс ожидающей сущностью такого же уровня
- RECEIVE (прием) – ожидание входящего сообщения
- SEND (передача) – отправка сообщения сущности того же уровня
- DISCONNECT (разрыв) – разрыв соединения

Проблемы при разработке стека протоколов

Некоторые из ключевых аспектов разработки, возникающих при создании компьютерных сетей, присутствуют на нескольких уровнях.

Адресация

Каждый уровень нуждается в механизме идентификации отправителей и получателей, следовательно необходима система адресации.

Логические каналы

Протокол должен определять количество логических каналов, относящихся к соединению, а также их приоритеты. Так, многие сети обеспечивают как минимум два канала на соединение: один для обычных данных, другой – для срочных.

Контроль над ошибками

В протоколе должен быть предусмотрен контроль над ошибками. Получатель должен иметь возможность сообщить отправителю, какие из сообщений были получены неправильно.

Нумерация

Некоторые каналы связи (ориентированные на коммутацию пакетов) могут доставлять сообщения не в порядке отправления, поэтому протокол должен явно снабжать сообщение номерами, чтобы их можно было собрать в правильном порядке.

Управление потоком

Также возникает вопрос: что делать, чтобы более быстрая сторона не завалила своими пакетами более медленную ? Одно из решений – контроль состояния сторон, другое решение – договоренность по ограничению скорости передачи между сторонами. В целом это называется управлением потоком.

Размер сообщений

Еще одна проблема возникает с размерами сообщений: что делать с огромными сообщениями и очень малыми, которые неэффективно пересылать? Например – разбивать пакет на много малых либо наоборот.

Мультиплексирование

Иногда неэффективно или невозможно установить отдельное соединение для каждой пары общающихся процессов, тогда нижестоящий уровень может принять решение использовать одно и то же соединение для передачи нескольких соединений несвязанных между собой. Это называется уплотнением каналов или мультиплексированием и происходит прозрачно для вышестоящих уровней.

Маршрутизация

Бывает так, что между отправителем и получателем существует несколько путей следования сообщений, в этом случае возникает проблема выбора оптимального пути. Эта задача маршрутизации.

Спасибо за внимание!
Вопросы?