Распределенные информационные системы

Время

Вопросы

- Понятия мирового, системного и физического времени. Их взаимосвязь и зависимость
- Синхронное и согласованное выполнение операций в РИС
  - Методы синхронизации
  - Методы согласования
- Системы реального времени
- ОС реального времени

Мировое

Системное

Физическое

ВРЕМЯ В РИС

Понятие времени

- Форма протекания физических и психических процессов, условие возможности изменения
- Одно из основных понятий философии и физики, условная сравнительная мера движения материи, а также одна из координат пространства-времени, вдоль которой протянуты мировые линии физических тел
- The fourth dimension and a measure in which **events can be ordered** from the past through the present into the future, and also the **measure of durations of events and the intervals** between them

Практические свойства времени

- Однонаправленность
- Связано с событиями, значениями, объектами
- Измеримость:
  - Текущего момента
    - Временн*ая* метка
  - Длительностей процессов и интервалов между событиями
    - Отрезок, ограниченные двумя временн*ы*ми метками
- ~~Постоянно не хватает~~

Виды времени: мировое – 1/5

- Мировое время (wallclock time)
  - Текущее время в мире (неделимо до 10^{-44})
  - Измеряется и выдается потребителям несколькими мировыми организациями (дискретно!)
- Варианты мирового времени:
  - Полученное астрономическими средствами:
    - Среднее солнечное время (Mean Solar Time – UT1)
    - Всемирное координированное время (Coordinated Universal Time – UTC)
    - Эфемеридное время (Terrestrial (Ephemeris) Time – TT)
  - Полученное как результат работы средств измерения времени:
    - Международное атомное время (Temps Atomique International – TAI)
    - Системное время GPS (GPS), ГЛОНАСС, etc.

Врезка: Дополнительные секунды – 1/4

- Дополнительная секунда (leap second) — секунда, добавляемая к UTC для согласования его с UT1
  - Вводится по объявлению Международной службы вращения Земли (sic!) согласно астрономическим наблюдениям
    - Добавляется в конце суток по UTC 30 июня или 31 декабря так, чтобы UTC не отличалось от UT1 более, чем на ±0,9 секунды
      - В такие дни после времени 23:59:59 идёт 23:59:60
    - Теоретически возможно объявление отрицательной дополнительной секунды, если средние солнечные сутки окажутся короче календарных
      - В такие дни после 23:59:58 идет 00:00:00
      - На практике отрицательные дополнительные секунды никогда не объявлялись

Врезка: Дополнительные секунды – 2/4

- Причиной возникновения дополнительных секунд является замедление вращения Земли из-за:
  - Приливного замедления
  - Изменения формы Земли

  → разница между средними солнечными сутками и сутками в СИ (составляющими ровно 24 часа = 24*60*60 с)
  - в среднем растёт с ускорением примерно 64 секунд/век

Врезка: Дополнительные секунды – 3/4

- Учет дополнительных секунд:
  - Автономные часы учитывать их не умеют, так как поправка вводится по результатам астрономических наблюдений и заранее запрограммировать расписание невозможно
  - Если есть внешняя синхронизация:
    - Протокол синхронизации времени NTP информирует о наступающей доп. секунде
    - GPS передают уведомление о наступающей доп. секунде
    - Протокол NMEA (GPS-приёмники) передаёт информацию только о наступившей доп. секунде

Врезка: Дополнительные секунды – 4/4

- Разработчики GPS ввели собственную временную шкалу, синхронизировав время с UTC один раз в 1980 году при запуске → постоянная разница между временем GPS и TAI составляет 19 секунд (по состоянию на 2020 год)
- Проектировщики ГЛОНАСС используют UTC и его корректировки

Виды времени: системное – 2/5

- Системное время (system time)
  - Дискретное время хоста, на котором выполняется процесс (физические часы)
  - Дискретное время ОС (логические часы)
- Доступно через вызовы ОС или через специализированные средства доступа к таймеру
- [Локальные] физические часы могут быть:
  - Встроенным таймером материнской платы (с батарейкой / аккумулятором)
  - Отдельным устройством, возможно, подключенным к источнику точного времени
  - Приемником времени от источника точного времени и/или синхроимпульсов

Виды времени: физическое – 3/5

- Физическое время (physical time)
  - Для моделирующих систем: время в моделируемой системе
  - Для любых РИС: время во взаимодействующей реальной системе или устройстве (например, датчике)
- Доступно либо непосредственно из модели или через интерфейс с внешней системой. Чаще получается как указание, к какому моменту относятся те или иные данные

Виды времени: модельное – 4/5

- Модельное время (simulation time)
  - Время модели – момент времени, на который хранится (рассчитывается) набор значений моделируемых характеристик
  - Является дискретным представлением физического времени моделируемых объектов, систем, мира в целом
  - Процесс изменения модельного времени ***может*** быть привязан к изменению мирового времени или нет

Виды времени: модельное – 5/5

- Продвижение модельного времени может:
  - Имитировать движение мирового времени, если моделируется нечто, связанное с ним (положение космических тел, освещенность, погодные условия, etc.)
  - Служить для упорядочения моделируемых событий, которые зависят только от других моделируемых событий, но не зависят от мирового времени

В конечном итоге, все подобные зависимости сводятся к привязке именно к астрономическим событиям

Время: взаимосвязь видов

- Значения различных видов времен не равны
- Различные виды мирового времени подстраиваются друг к другу
- Модельное время может быть привязано к:
  - Мировому
  - Системному
  - Физическому
- Физическое время может быть привязано к мировому (см. синхронизация)

Время: нарушения свойств – 1/2

- Все доступные в ПО (и человеку вообще) виды времен дискретны
- Не непрерывны:
  - Некоторые виды мирового времени – для соответствия физическому смыслу
  - Модельное время – как следствие назначения моделирования
- Могут изменять скорость продвижения:
  - Модельное – зависит от решаемой задачи
  - Системное – см. NTP
  - Физическое – результат изменений в таймере

Время: нарушения свойств – 2/2

- Могут двигаться назад:
  - Модельное
  - Системное – см. NTP
  - Физическое – см. Синхронизация
  - Мировое – при подстройке под глобальные явления, к которым привязано

Врезка: Средства изменения времени

- Самое большое число движущихся частей - у песочных часов, а самые большие движущиеся части - у солнечных

Определения
Методы синхронизации
Методы согласования

СИНХРОННОЕ И СОГЛАСОВАННОЕ ВЫПОЛНЕНИЕ ОПЕРАЦИЙ

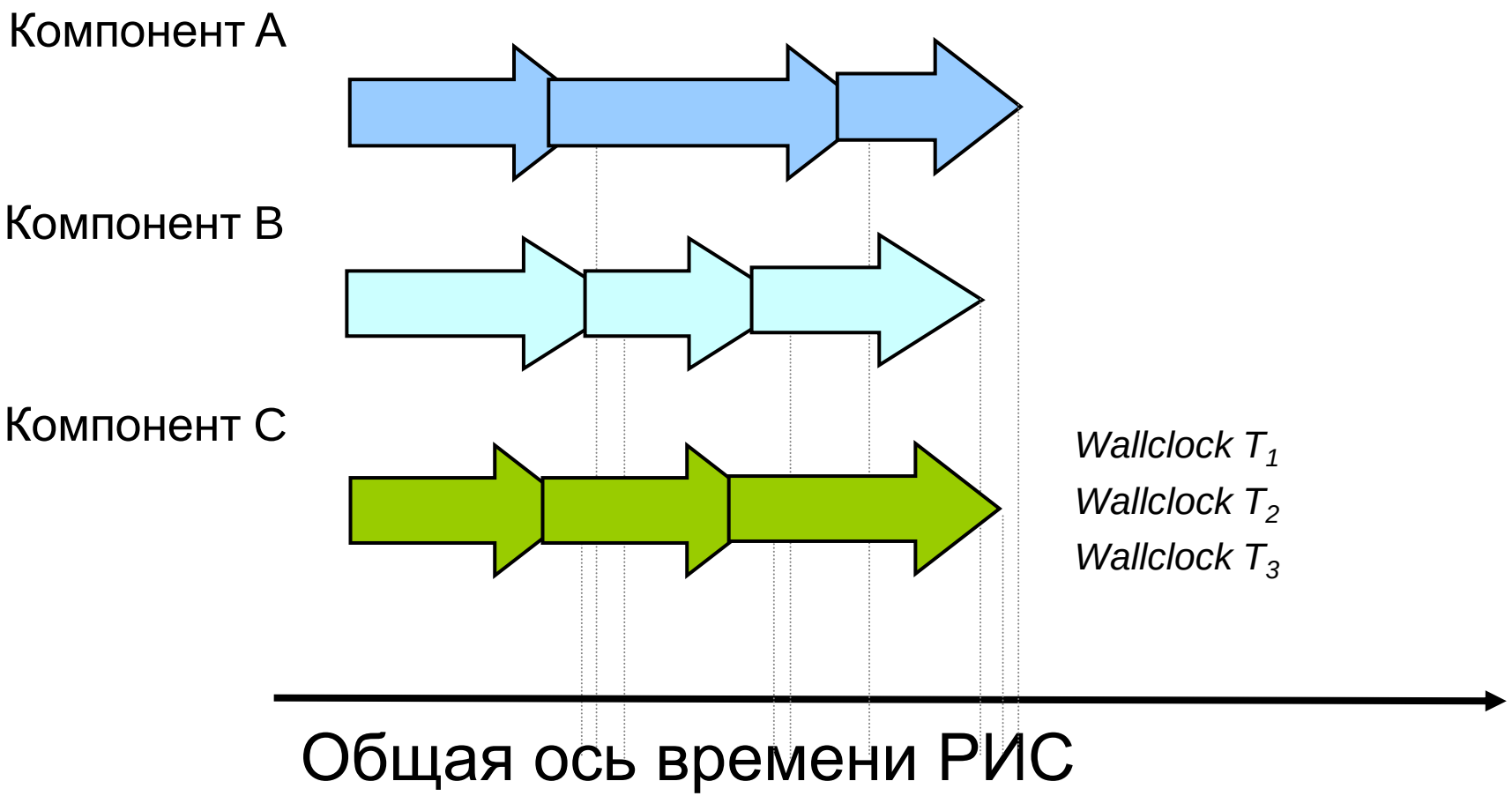
Время: продвижение

- Увеличением на некоторое значение
  - Название: time-stepped mechanism
  - Значение шага может быть фиксировано
  - Пример: полетный симулятор
- Приемом событий, обладающих характеристикой «момент времени» (time-stamp)
  - Название: event-driven mechanism
  - Продвижение локального времени полностью зависит от внешних событий
  - Пример: модель связного оборудования

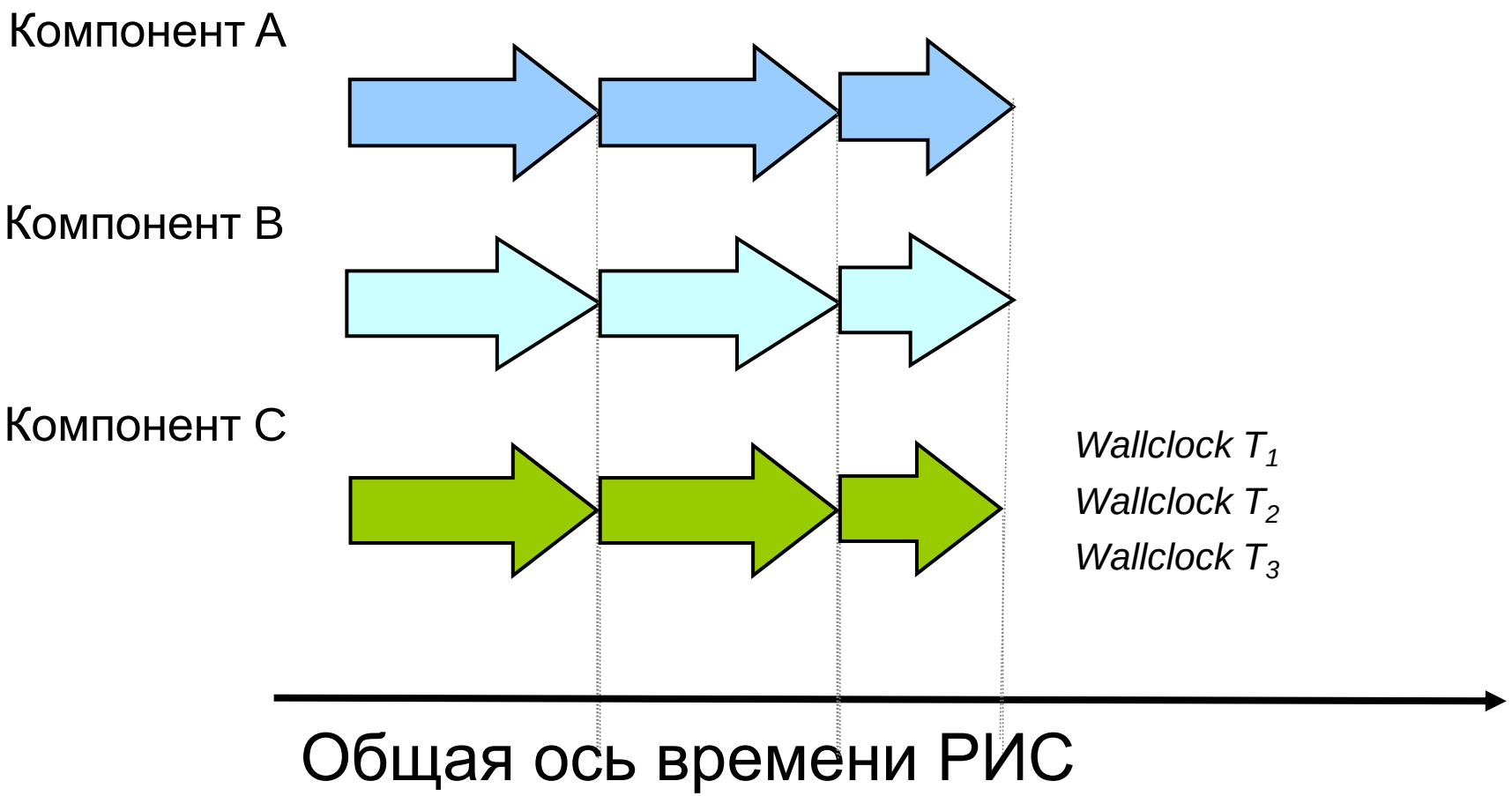
Синхронность и согласованность – 1/3

- Время в элементах РИС может продвигаться:
  - Синхронно
    - У всех компонентов РИС время равно
      - В каждый конкретный момент времени
      - В заданные моменты мирового времени (обычно назначаемые через одинаковые интервалы)
  - Согласованно
    - Расхождение времен компонентов РИС не влияет на функционирование РИС в целом

Синхронность и согласованность – 2/3



Синхронность и согласованность – 3/3



Необходимость синхронизации

- Необходимо воспроизводить причинную последовательность событий
  - Пример: выстрел из орудия должен предшествовать попаданию в цель
- Многие модели и системы не могут принимать сообщения «из прошлого»
- Требуется повторяемость цепочки событий (прогона моделирования)
  - Детерминистические модели должны выдавать одинаковые результаты при каждом выполнении

Методы синхронизации

- Синхронизация системного или физического времени:
  - Протокол NTP
  - Протокол SPT (поверх MIL-BUS 1553)
  - IRIG-B
- Синхронизация модельного времени:
  - Сложно реализуема
  - Нестандартизована
  - Требует реализации этой функциональности в ПО компонентов РИС или разработки специализированного middleware

Методы согласования

- Согласуется ***только*** модельное время и/или выполнение распределенных операций РИС
- Методы согласования:
  - [распределенное моделирование] Группа сервисов Управления логическим временем стандарта IEEE-1516
  - [чаще, универсально] Уникальная схема для конкретной РИС

Временные метки

- Могут назначаться:
  - Событиям
  - Изменениям значений характеристик объектов
- Могут содержать:
  - Модельное время
  - Системное время
  - Физическое время
- Используются для
  - Журналирования
  - Синхронизации или согласования выполнения

Протокол NTP
Протокол SPT
Технология / протокол IRIG-B

СИНХРОНИЗАЦИЯ СИСТЕМНОГО (ФИЗИЧЕСКОГО) ВРЕМЕНИ

Технологии синхронизации

Технология	Что синхронизует	Как
NTP	Системное время ОС	Процессы ОС через IP сеть обмениваются данными о текущем времени Процессы подстраивают время ОС «как надо»
IRIG	Физические таймеры оборудования	Источник времени передает синхросигналы заданной формы через спец. кабели
SPT	Время устройств	Протокол обмена включает посылку сигналов в заданные моменты времени

ПРОТОКОЛ NTP

Протокол NTP – основные сведения

- Название: Network Time Protocol
- Один из старейших:
  - Создан в 1985
  - Развивается: Текущая (на 2020) версия — NTP 4 (от 2010 года)
- Назначение протокола: Синхронизация системного времени к UTC
- Спецификация: RFC 5905
- Сайт: http://ntp.org/

Состав системы NTP

- Основой является **алгоритм** получения меток времени через сеть IP, что породило:
  - Набор приложений, этот алгоритм реализующих
  - **Протокол** обмена между этими приложениями поверх UDP

→ Иерархическая **РИС**, свободно доступная через Интернет по протоколу NTP

Врезка: Авторы NTP

- Разработан (спроектирован) в целом Дэвидом Л. Миллсом (David L. Mills) из университета Дэлавера

- Расчетная часть NTP использует модифицированный алгоритм Марзулло (Кит Марзулло - Keith Marzullo) из Университета Калифорнии

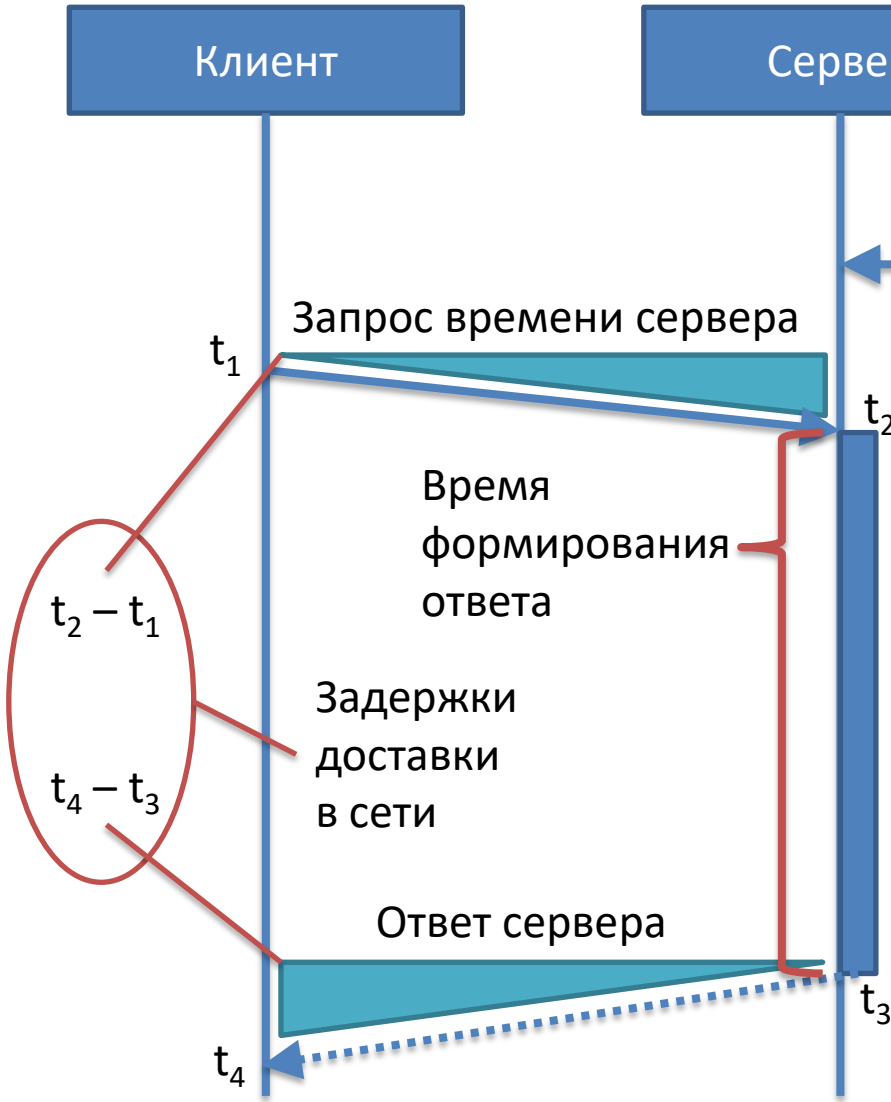
Алгоритм NTP – 1/5

- Устойчив к:
  - Изменениям задержки сигнала в сети
  - Потерям, повторениям и spoofing пакетов
- Точность версии 4:
  - Интернет: до 10 мс
  - ЛВС: 200 мкс и лучше

Алгоритм NTP – 2/5

- Расчет смещения времени относительно сервера производится путем опроса (polling) сервера:
  - Клиент шлет запрос на получение времени от сервера
  - Сервер в ответ присылает значение текущего времени сервера
  - Каждая посылка получает две временные метки (отправки и получения). Три из них возвращаются клиенту в ответе сервера

Алгоритм NTP – 3/5



1. Запоминается время приема запроса t_2
2. Разбирается пакет запроса: Запоминается время отправки запроса t_1
3. Составляется пакет ответа. Записываются:
   1. Время обновления сервера
   2. Время отправки запроса t_1
   3. Время приема запроса t_2
   4. … другие поля
   5. «Время отправки» ответа t_3

NB! В Википедии времена нумеруются с «0», в RFC – с «1»

Алгоритм NTP – 4/5

- Разница во времени [между таймерами] сервера и клиента ([clock] offset):

$$\theta = \frac{(t_2 - t_1) + (t_3 - t_4)}{2}$$

- «Круговая» задержка доставки ([round-trip] delay):

$$\delta = (t_4 - t_1) - (t_3 - t_2)$$

Алгоритм NTP – 5/5

- Значения Θ и δ используются для набора статистики работы алгоритмов «смягчения» (mitigation)
- Времена t_1 и t_3 – определяются программным образом (softstamps) и их значения могут «плавать»

Протокол NTP – 1/4

- Уровень (по модели OSI): Прикладной
  - Семейство: TCP/IP
  - Порт/ID: 123/UDP
- Структура пакета описана в RFC 5905
  - Пакет состоит из целого числа 32-битных слов
  - Заголовок состоит из 48 байт (12 слов)
- Клиент-серверный, но может использоваться для построения peer-to-peer сетей, где каждый участник рассматривается как источник точного времени

Протокол NTP – 2/4

- Может включаться режим multicast или broadcast, когда клиенты пассивно получают обновления времени после начального калибровочного обмена
- Предусматривает рассылку предупреждения о предстоящем введении дополнительной секунды
- Не предусматривает рассылку информации о временных поясах и летнем времени

Протокол NTP – 3/4

- Представление времени в NTP:
  - Короткий (NTP short format)
    - 32 бита (16 бит – секунды, 16 бит – доли секунды)
    - Для указания задержек и дисперсии
  - Временн***а***я метка (NTP timestamp format)
    - 64 бита (32 бита – секунды, 32 бита – доли секунды)
    - Минимальная доля 2^{-32} = 232 пикосекунды
    - Максимально поле секунд содержит 136 лет
    - Отсчитывается с 1 января 1900 года (NB! Не с 1970)
    - Для определения даты получатель должен хотя бы примерно (с точностью 68 лет) знать текущее время

Протокол NTP – 4/4

- Время в NTP (продолжение):
  - Дата (NTP Date format)
    - 128 бит:
      - 32 бита – Номер эры
      - 32 бита – Сдвиг в эре (Era offset) – секунды с начала эры
      - 64 бита – Доли секунд
    - Эра 0 начинается с 1 января 1900 года
    - Эра 1 начнется в 2036 году (предположительно 8 февраля)

Врезка: Интересные даты NTP

Дата	MJD	NTP era	NTP Offset	Что это?
1 января -4712	-2,400,001	-49	1,795,583,104	Первый день Юлианского календаря
1 января -1	-679,306	-14	139,775,744	2 год д.н.э.
1 января 0	-678,491	-14	171,311,744	1 год д.н.э.
1 января 1	-678,575	-14	202,939,144	1 год н.э.
4 октября 1582	-100,851	-3	2,873,647,488	Последний день Юлианского календаря
15 октября 1582	-100,840	-3	2,874,597,888	Первый день Григорианского календаря
31 января 1899	15,019	-1	4,294,880,896	Посл. день NTP эры -1
1 января 1900	15,020	0	0	Первый день NTP эры 0
1 января 1970	40,587	0	2,208,988,800	Первый день UNIX
1 января 1972	41,317	0	2,272,060,800	Первый день UTC
31 декабря 1999	51,543	0	3,155,587,200	Последний день 20 века
8 февраля 2036	64,731	1	63,104	Первый день NTP эры 1

Врезка: Календари – 1/4

- Календари:
  - Год – промежуток между повторяющимися ежегодно событиями:
    - Зимнее солнцестояние (см. Гозекский круг)
    - Восходы Сириуса (см. Древний Египет)
    - Весеннее равноденствие (см. Древний Рим)
  - Отсчет годов ведется от значительных событий:
    - Сотворения мира
    - Основания Рима
    - Вступления на трон первого царя
    - Начала правления новой династии
    - Рождества Христова
    - Хиджры

Врезка: Календари – 2/4

- Самые известные календари:
  - Юлианский (устарел)
    - Введен Юлием Цезарем 1 января 45 года д.н.э.
    - Действовал в России до 14 февраля 1918 («старый стиль»)
    - Использовался в Греции до 1924, в Турции до 1926 и Египте до 1928
  - Григорианский (действует)
    - Введен папой римским Григорием XIII 4 октября 1582
    - Основной международный календарь
    - Точнее совпадает с солнечным годом
    - Последними перешли Китай в 1949 и Саудовская Аравия в 2016

Врезка: Календари – 3/4

- Также действующие календари:
  - Календарь Хиджры (действует)
    - Отсчет ведется от Хиджры (16 июля 622 года н. э.) — даты переселения пророка Мухаммеда и первых мусульман из Мекки в Медину
    - Лунный: 12 месяцев, 354 или 355 дней
    - Сутки начинаются в момент захода солнца
    - Месяц начинается в день, когда серп Луны можно видеть в вечерние сумерки впервые после новолуния (обычно через 1…3 дня после новолуния)
    - Месяцы быстро(!) сдвигаются относительно григорианского календаря
    - Используется мусульманами

Врезка: Календари – 4/4

- Также действующие календари:
  - Буддийский (действует)
    - Лунно-солнечный:
      - Начало года всегда приходится на декабрь
      - Месяцы привязаны к фазам луны
    - Для соответствия лунных и солнечных периодов, периодически вводятся дополнительные дни и дополнительные месяцы (sic!) → год может длиться 354, 384 или 385 дней (либо 354, 355 и 384)
    - Опережает григорианский календарь на 543 года (2020 соответствует 2563)
    - Используется в Таиланде, Лаосе, Камбодже, Мьянме и на Шри-Ланке

Врезка: Юлианский период – 1/2

- Юлианский период
  - Используется для целей истории и хронологии
  - Год нумеруется тремя числами — индиктом (от 1 до 15), лунным циклом (от 1 до 19) и солнечным циклом (от 1 до 28)
  - Цикл начинается 1 января 4713 до н. э. (все числа были равны 1
  - Длительность цикла в годах равняется 15·19·28 = 7980 лет
  - Конец первого юлианского периода придётся на 23 января 3268 года (по григорианскому календарю – текущему «общечеловеческому»)

Врезка: Юлианский период – 2/2

- Юлианская дата (JD) – количество суток, прошедших начиная с **полудня** понедельника 24 ноября 4714 г. до н. э.
  - Первый день имеет номер 0
  - Даты сменяются в полдень UT или TT
- Модифицированная юлианская дата (Modified Julian Day – MJD)
  - MJD = JD - 2400000.5
  - Отсчитывается от **0 часов** 17 ноября 1858 года
  - Предложена в 1957 году Смитсоновской астрофизической обсерваторией

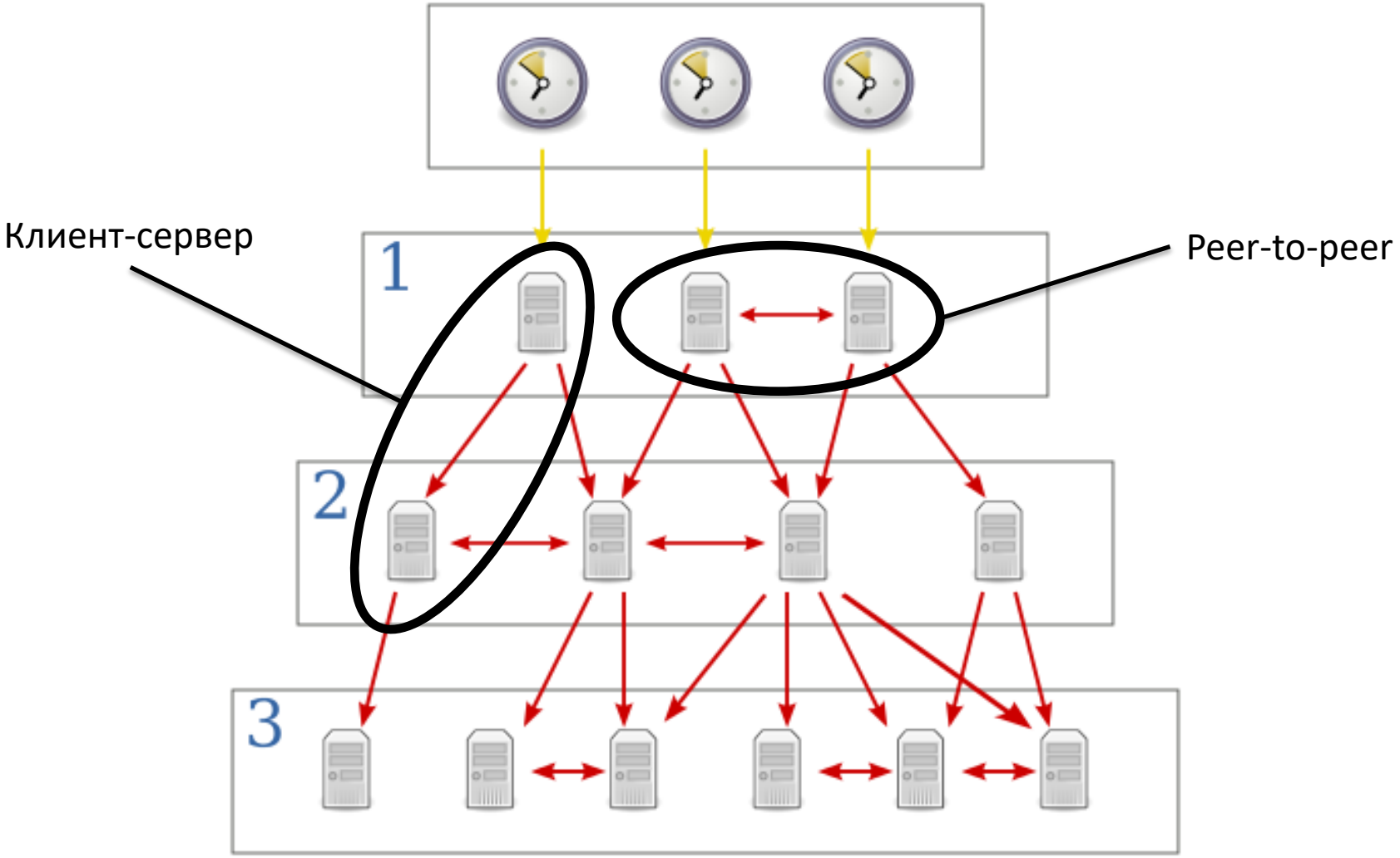
Clock strata – Часовые уровни – 1/6

- NTP использует иерархическую, «полуслойную» систему связанных серверов времени
- Уровень иерархии называется стратой (лат. stratum – уровень). Ему присваивается номер, начинающийся с нуля
- Нулевым уровнем считаются средства измерения времени (опорные часы)
- Всего может быть 15 слоев. Нахождение в слое 16 интерпретируется как несинхронизованное состояние устройства

Clock strata – Часовые уровни – 2/6

- Сервер, синхронизированный с сервером уровня n, работает на уровне n + 1
  - Номер уровня численно представляет расстояние от опорных часов и используется для предотвращения циклических зависимостей в иерархии
  - Слой не всегда является показателем качества или надежности
- Можно найти свободно доступные сервера уровня 3 и даже уровня 2

Clock strata – Часовые уровни – 3/6



Clock strata – Часовые уровни – 4/6

- Слой 0
  - Высокоточные приборы, служащие эталоном времени (reference clocks), такие как атомные (молекулярные, квантовые) часы, радиочасы или их аналоги
  - Обычно эти устройства не имеют сетевого интерфейса. Они подключены к локальному компьютеру и передают сигналы PPS для синхронизации (см. протокол IRIG), порождая прерывание (см. ОС реального времени)
  - Типичные представители слоя 0: GPS, ГЛОНАСС, атомные эталоны времени

Clock strata – Часовые уровни – 5/6

- Слой 1
  - Компьютеры, к которым напрямую подключены эталонные часы. Их называют первичными серверами времени
  - Выступают в качестве сетевых серверов времени и отвечают на NTP-запросы, посылаемые компьютерами слоя 2
  - Могут соединяться с другими серверами слоя 1 для проверки работоспособности и резервного копирования

Clock strata – Часовые уровни – 6/6

- Слой 2 и более низкие
  - Компьютеры, получающие по протоколу NTP время от серверов первого (предыдущего) слоя
  - Обычно подключаются одновременно к нескольким серверам, и используя NTP-алгоритм:
    - Получают наилучший образец данных
    - Отсеивают сервера с очевидно неверным временем
  - Могут сравнивать свои данные с другими компьютерами своего слоя для получения стабильных и непротиворечивых данных
  - Являются серверами для следующего слоя

Реализации NTP – 1/3

- Существует несколько реализаций NTP:
  - Эталонная (reference) – см. ниже
  - SNTP (Simple NTP)
    - Не включает долгого хранения состояния
    - Используется во встроенных системах
  - Windows Time
    - Периодически «прыжком» синхронизует время
  - chrony
    - Предназначения для нестабильных «засыпающих» систем, включая виртуальные машины
  - и другие…

Реализации NTP: Эталонная – 2/3

- Эталонная (reference)
  - Развивается более 20 лет совместно с протоколом / алгоритмом
  - Портирована на множество платформ, включая Unix и Windows
  - Была подробно проверена в 2017 с указанием множества потенциальных дыр в безопасности
  - После выхода на пенсию Девида Миллса поддерживается как проект open-source Харланом Стенном (Harlan Stenn)

Реализации NTP: Эталонная – 3/3

- Включает несколько приложений:
  - `ntpdate` – скачком приводит системное время к точному
  - `ntpd` – сдвигает системное время к точному плавно. «Догоняет» точное время достаточно быстро. «Дожидается» точное время относительно долго
    - Предпочтительно запускать `ntpd` как системную службу (Windows) или демон (Unix/Linux)
  - `ntpq` – терминальное приложение, позволяющее мониторить состояние системы синхронизации, вводя запросы в окне приложения
- Реализация ntp по умолчанию входит в дистрибутив Unix/Linux, обычно в «обертке» графического интерфейса настройки и мониторинга состояния

ПРОТОКОЛ SPT

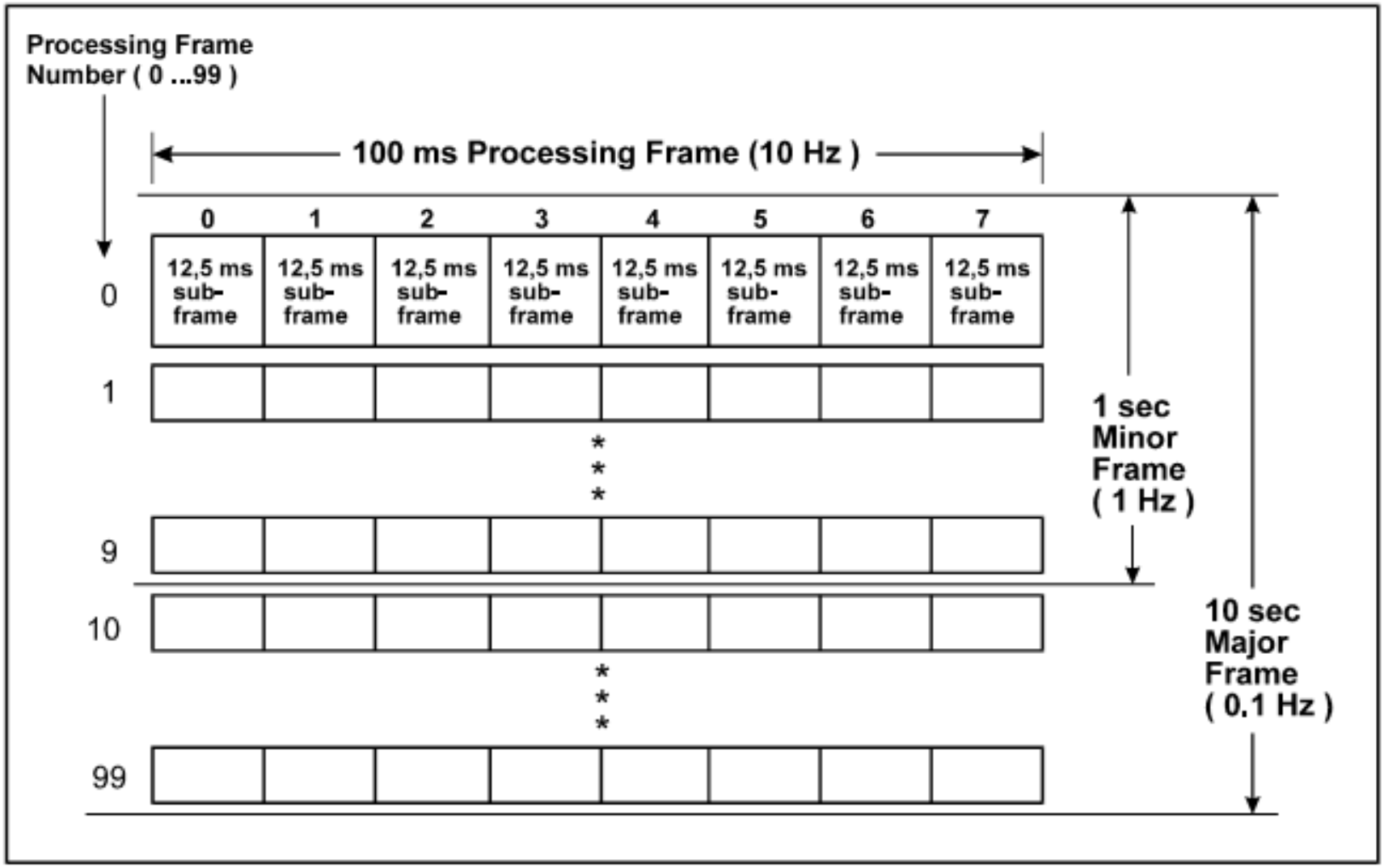
Стандарт MIL-STD-1553

- Стандарт МО США (US DoD) последовательной шины передачи данных
- Изначально был разработан для управления авионикой, сейчас широко используется для бортовых систем гражданского и военного назначения
- Число абонентов шины:
  - 1 контроллер шины (Bus Controller, BC)
  - 31 подчиненное устройство (Remote Terminal, RT)
- Стандарт MIL-STD-1773 описывает аналогичную оптическую шину

Протокол SPT – 1/3

- Название: MIL-Bus Synchronous Packet Transfer
- Особенности:
  - Циклический
  - Синхронизованный
  - Использующий расписание
- Состоит (иерархически) из:
  - Major Frame (0.1 Гц)
  - Minor Frame (1 Гц)
  - Processing frame (10 Гц)

Протокол SPT – 2/3



Протокол SPT – 3/3

- Каждый Processing frame начинается с широковещательной посылки, содержащей номер этого Processing frame в Major Frame
- Каждый Major Frame начинается с посылки широковещательного сообщения, содержащего полную текущую дату и время
- Точность передачи составляет +/- 50µs

ПРОТОКОЛ IRIG

IRIG – 1/3

- IRIG (Inter-range instrumentation group time codes) – форматы кодирования информации о времени, используемые для ее распространения
- Источники точного времени (например, атомные осцилляторы, приемники GPS) часто оборудованы выводом IRIG
- 1960 – принята первая версия стандарта

IRIG – 2/3

- IRIG-B – TBD
- IRIG-H – 1 PPS – 1 импульс в секунду – передаются импульсы и метка времени
- 10 MHz – импульсы синхронизации

IRIG – 3/3

Внезапно – Ethernet для раздачи времени через NTP

1PPS

10 MHz

Внешний PPS

Вход от GPS антенны

«Жесткое» и «мягкое» реальное время

Коэффициент ускорения

СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Определения – 1/2

- Общие определения:
  - Система реального времени (СРВ) —реагирует на события во внешней по отношению к системе среде или воздействовать на среду в рамках требуемых временных ограничений
  - Real-time system – controls an environment by receiving data, processing them, and returning the results sufficiently quickly to affect the environment at that time

Определения – 2/2

- В моделировании:
  - Модельное время СРВ движется с той же скоростью, что и мировое время
- В системах обработки информации:
  - Результат выдается без различимой задержки

СРВ – общие замечания

- Характерное время задержки реакции – миллисекунды, иногда – микросекунды (бывают и сравнительно большие времена)
- СРВ часто путают с высокопроизводительными системами
- Существуют near real-time системы, в которых задержка обработки данных стабильна и они выдают результат с той же интенсивностью, что и получают входные данные, но результат «сдвинут»

Характеристики

- Дедлайн (deadline) — критический срок обслуживания, предельный срок завершения какой-либо работы
- Латентность (latency) — время отклика (время задержки) системы на внешние события
- Джиттер (jitter) — разброс значений времени отклика

Типы СРВ

- По допустимости нарушений временных ограничений:
  - Жёсткого реального времени (hard real-time)
    - Нарушения равнозначны отказу системы
  - Firm real-time
    - Нечастые нарушения позволяют СРВ функционировать, но могут привести к снижению качества работы
    - Полезность результата после дедлайна нулевая
  - Мягкого реального времени (soft real-time)
    - Нарушения приводят лишь к снижению качества работы системы

Коэффициент ускорения

- В РИС часто используется работа в режиме near real-time и/или с использованием коэффициента ускорения относительно мирового времени
- Коэффициент ускорения может быть:
  - Равен единице
  - Быть меньше единицы
  - Быть больше единицы

ОПЕРАЦИОННЫЕ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Определения – 1/3

- Операционная система реального времени (ОС РВ, real-time operating system, RTOS) — тип ОС, основное назначение которой — предоставление необходимого и достаточного набора функций для проектирования, разработки и функционирования систем реального времени на конкретном аппаратном оборудовании

Определения – 2/3

- The Single UNIX Specification, version 2:
  - Realtime in operating systems: the ability of the operating system to provide a required level of service in a bounded response time
  - Реальное время в операционных системах — это способность ОС обеспечить требуемый уровень сервиса в определённый промежуток времени

Определения – 3/3

- СРВ – аппаратно-программный комплекс, реагирующий в предсказуемые времена на непредсказуемый поток внешних событий. Следовательно:
  - Система должна успеть отреагировать (породить результат) на событие, за время не большее deadline

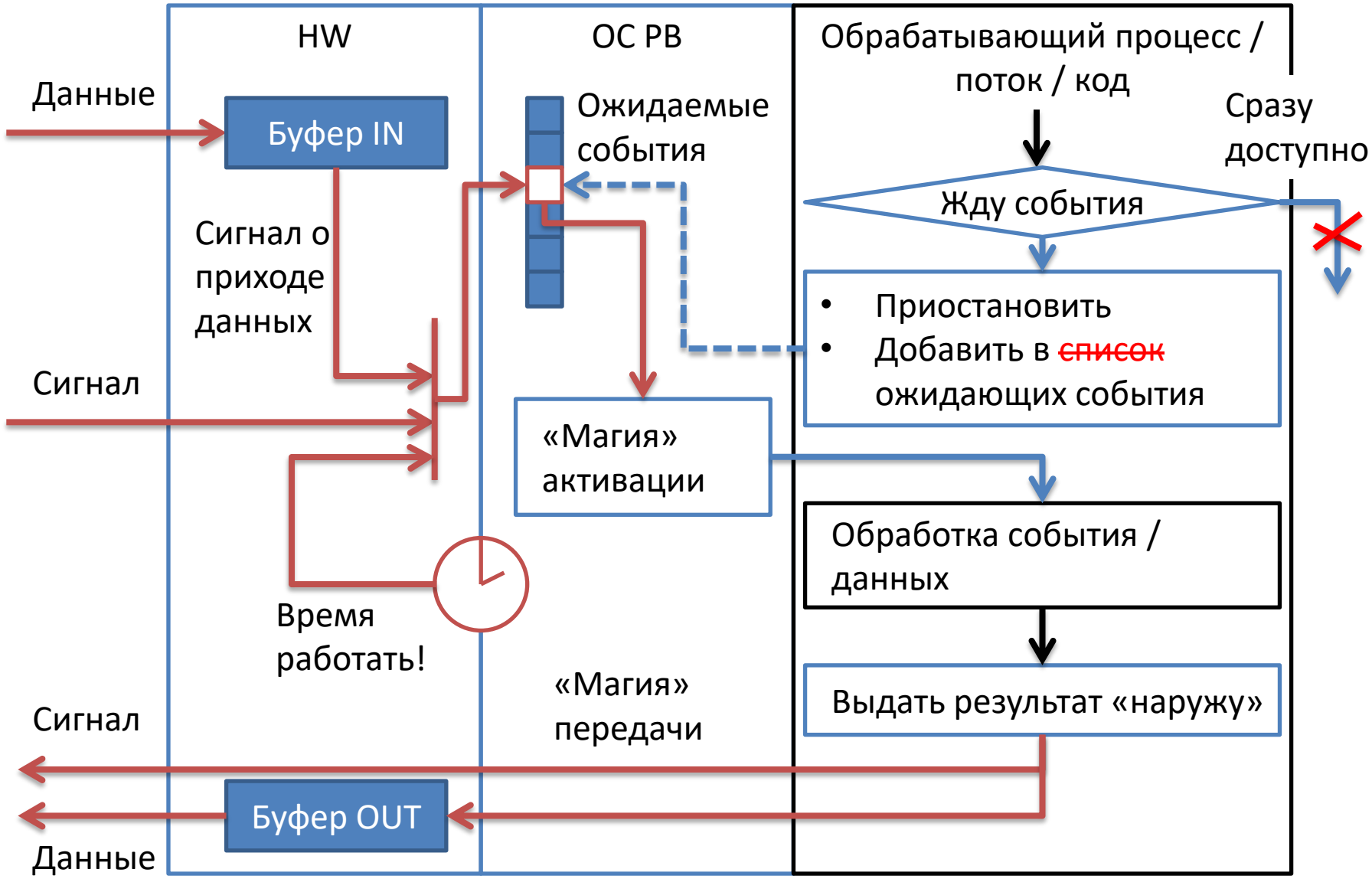
    Выполнимо
  - Система должна успевать реагировать на одновременно происходящие события

    Принципиально недостижимо в общем случае!

Как устроена задержка?

- Реакция СРВ на событие – процесс получения входных данных, их обработки и выдача полученного результата. Можно считать, что состоит из этапов:
  - Запись в буфер данных от внешнего источника
  - Порождение события о приходе данных (прерывание)
  - Активация обработчика прерывания (кода, считывающего входные данные и порождающего результат)
  - Запись результата в буфер, доступный внешним устройствам

Задержка в картинках



Задержка – подробнее – 1/2

- Приход данных, сигнал о [внешнем] событии и наступление момента времени преобразуются однотипно в сигнал о наступлении события
- Таймер может быть программным, аппаратным и их комбинацией
- Регистрация нескольких обработчиков одного и того же события для ОС РВ нехарактерна

Задержка – подробнее – 2/2

- Система в целом строится так, чтобы регистрация обработчиков событий происходила до фактического возникновения событий
  - Иначе принципиально невозможно гарантировать время реакции
- Активация обработчика в ОС РВ «быстрая» и отличается от «обычных» ОС
- Выдача результата потребителю – операция, по сути обратная обработке сигнала

Задержка и ОС РВ

- ОС РВ контролирует:
  - Задержку активации обрабатывающего кода по событию
  - Задержку выдачи результата вовне («делания» его доступным внешним устройствам)
- ОС РВ не контролирует:
  - Время обработки данных
- Как может влиять на обработку данных:
  - Ограничить максимальное время обработки
  - Отслеживать приоритеты (если обработка длится больше одного цикла активации кода)

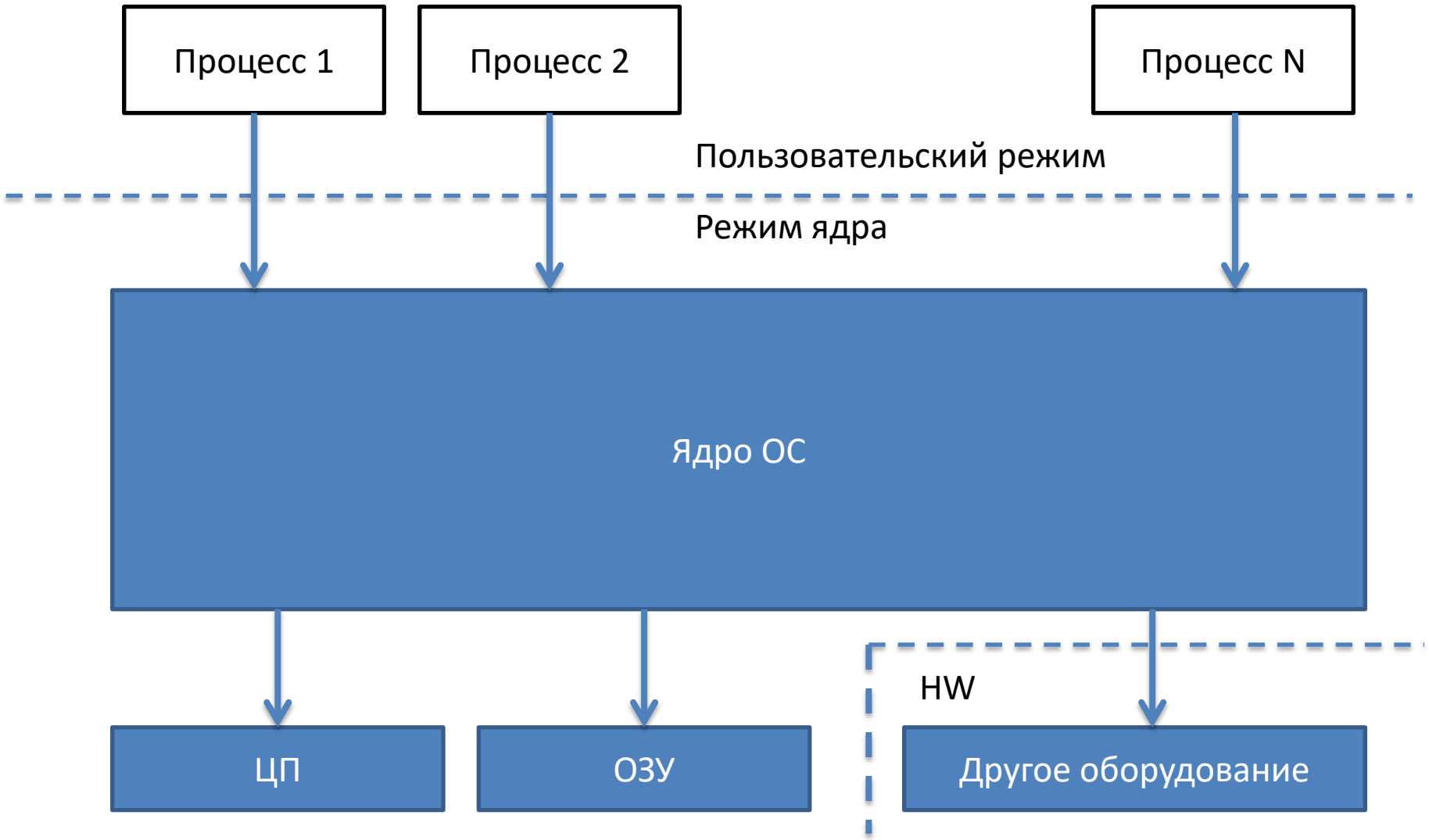
Классификация ОС РВ

- Основными архитектурами ОС РВ являются:
  - Монолитная
  - Модульная
  - Уровневая (слоевая)
  - Микроядерная («клиент-сервер»)
- По способу разработки СПО:
  - Self-hosted
  - Host/Target

Архитектура ОС РВ – монолитная – 1/4

- ОС РВ монолитной архитектуры обладает следующими особенностями:
  - ОС формируется как неделимый набор модулей ядра
  - Предоставляет СПО интерфейсы ко всему оборудованию через API ядра

Архитектура ОС РВ – монолитная – 2/4



Архитектура ОС РВ – монолитная – 3/4

- Преимущество:
  - Относительная быстрота работы. Однако, достигается это, в-основном, за счет написания значительных частей ОС на ассемблере

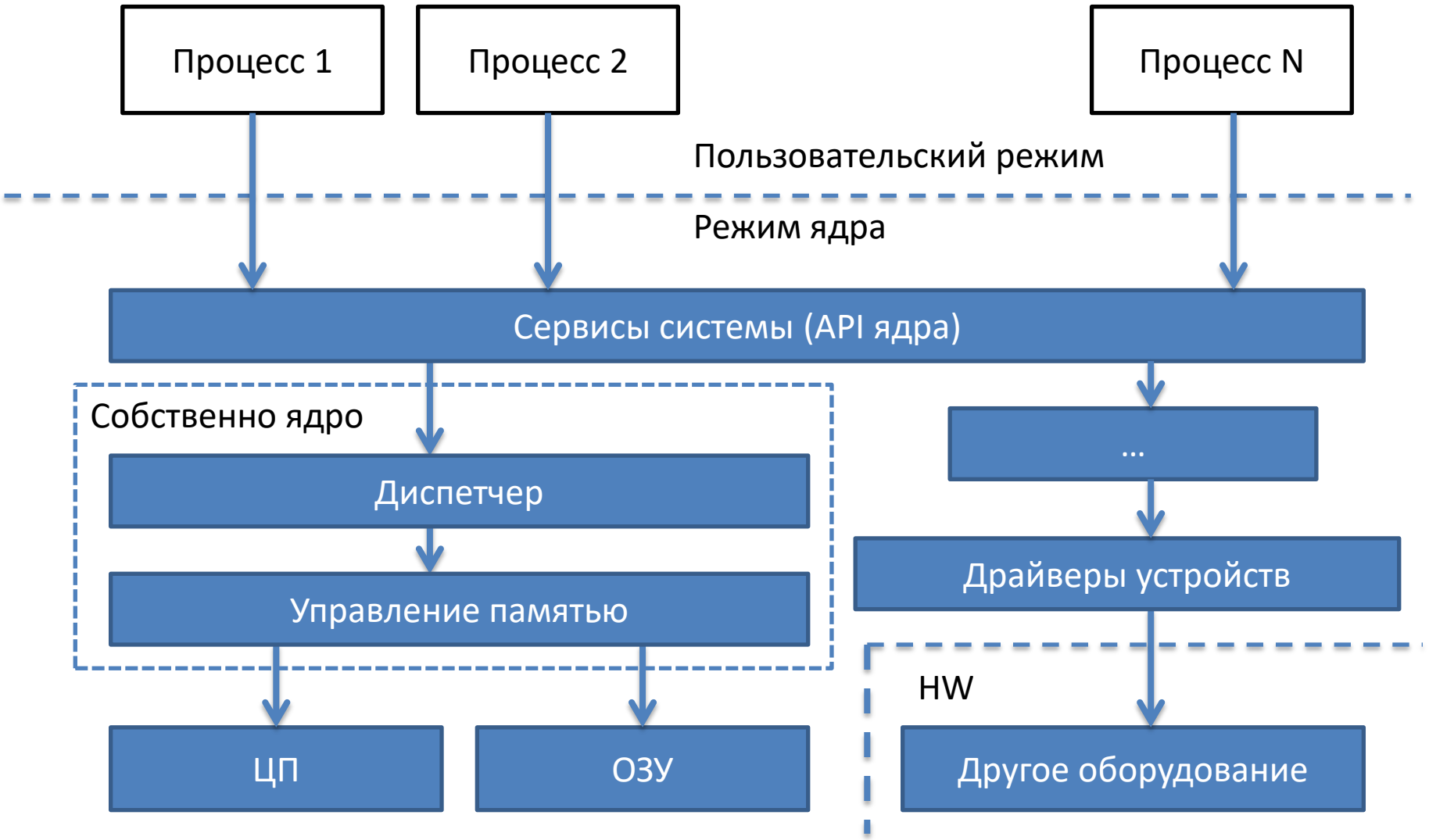
Архитектура ОС РВ – монолитная – 4/4

- Недостатки:
  - Ядро не может быть прервано СПО (non-preemptable), при этом все вызовы API выполняются в ядре → Может приводить к «захвату» ЦП низкоприоритетной (НП) задачей
    - Например, НП задача запросила выделение памяти через системный вызов → до окончания этого вызова сигнал активации ВП задачи не сработает
  - Сложность переноса на новые архитектуры процессора из-за значительных ассемблерных вставок
  - Изменение части ядра требует его полной перекомпиляции

Архитектура ОС РВ – модульная – 1/3

- Модульная архитектура отличается от монолитной структурой ядра:
  - Функции ядра организуются в иерархию модулей, имеющих «северный» и «южный» интерфейсы
  - В некоторых реализациях ОС многослойность реализуется за счет иерархии режимов процессора

Архитектура ОС РВ – модульная – 2/3



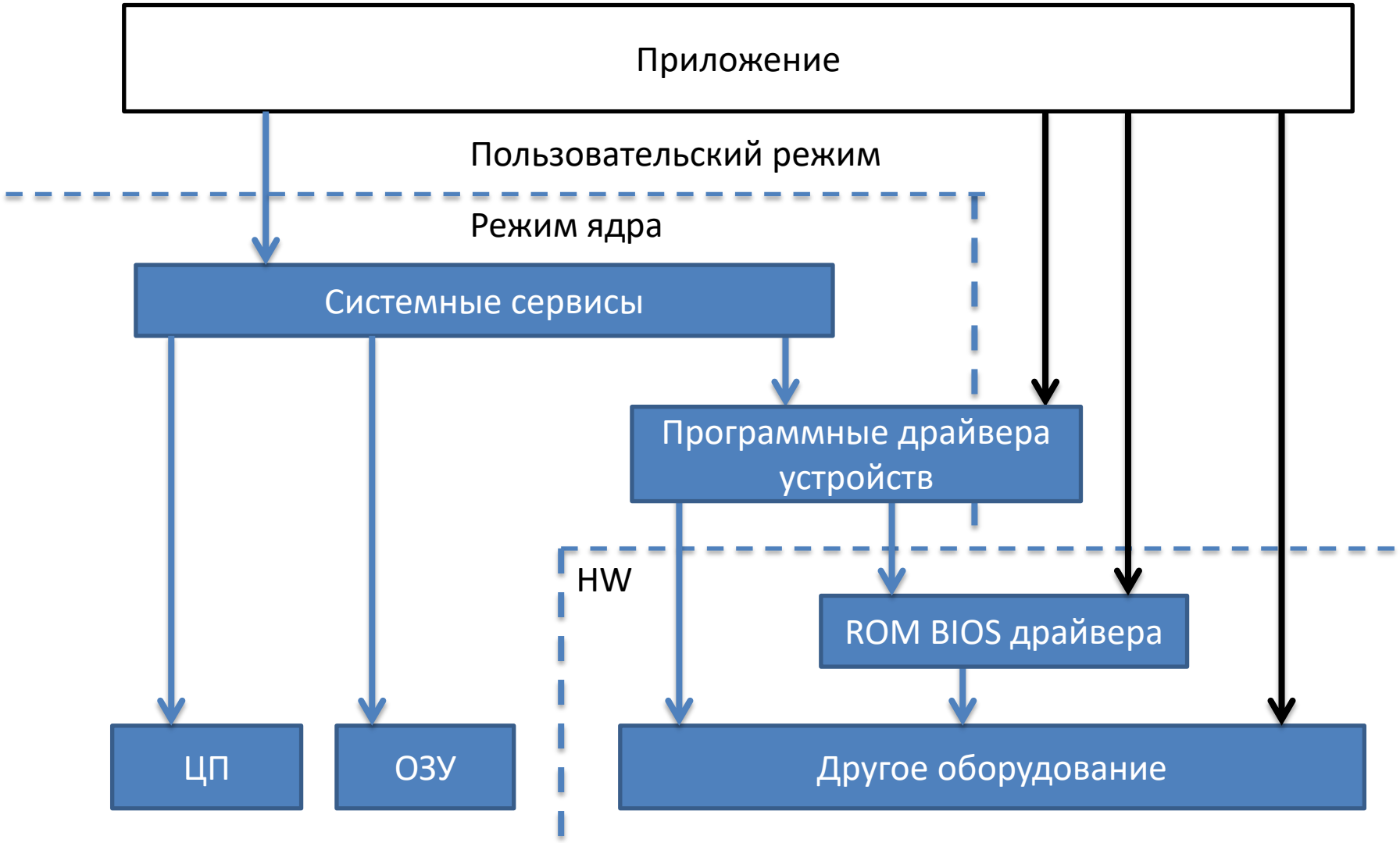
Архитектура ОС РВ – модульная – 3/3

- Преимущества:
  - Упрощение отладки ОС
  - Возможность замены реализаций слоев независимо от других
- Недостатки:
  - Непрерываемость функций ядра (аналогично монолитной архитектуре)

Архитектура ОС РВ – уровневая – 1/3

- Уровневая архитектура:
  - Основные сервисы ОС формируют ядро, но часть модулей доступна к аппаратуре находится вне ядра
  - СПО имеет возможность получить доступ к аппаратуре не только через ядро ОС, но и напрямую
  - СПО организуется в виде единственного процесса. Максимальный вариант многозадачности – потоки в этом процессе

Архитектура ОС РВ – уровневая – 2/3



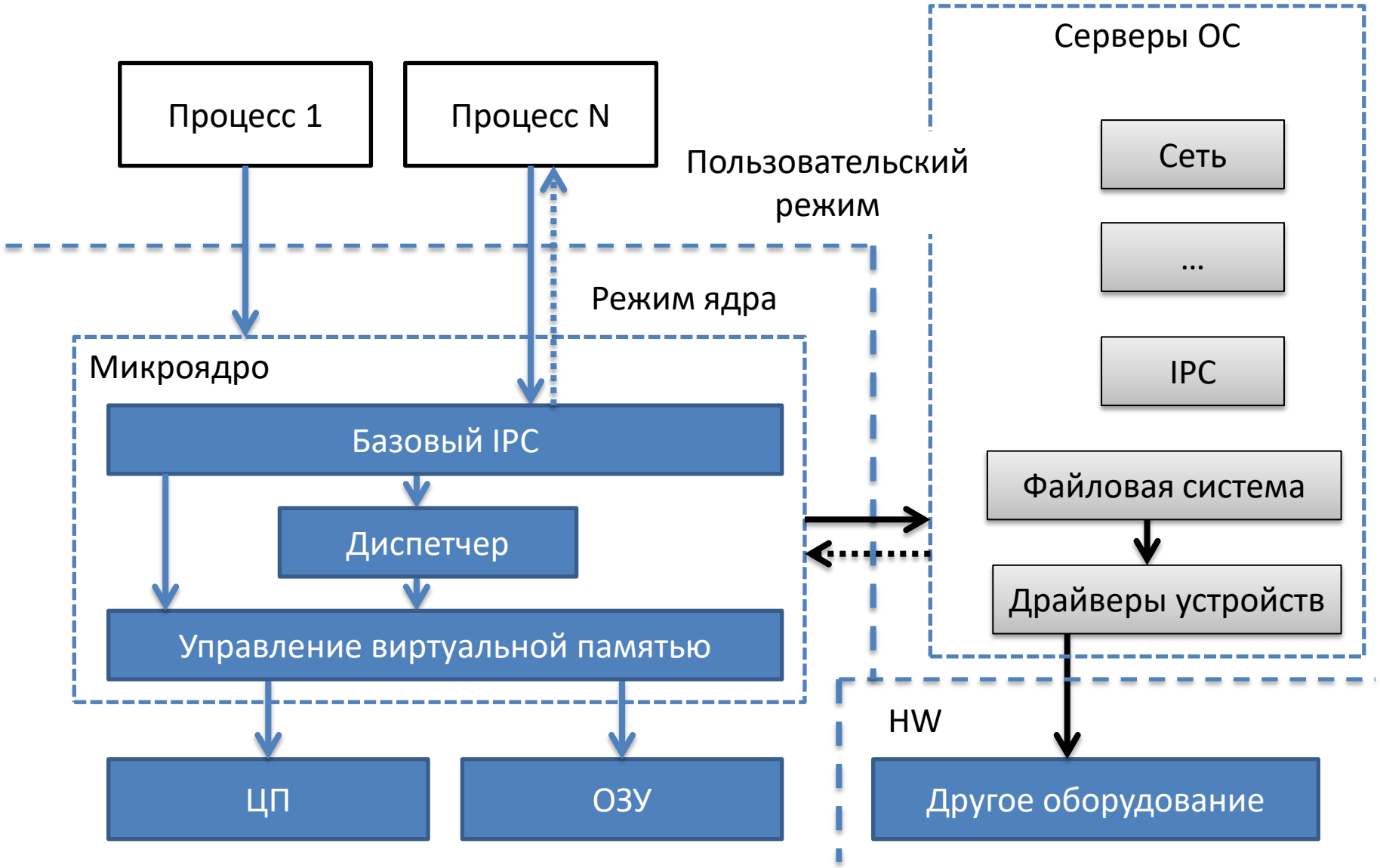
Архитектура ОС РВ – уровневая – 3/3

- Преимущества:
  - Нет переключения процессов, а только потоков (меньше накладные расходы) → высокая скорость
  - Меньше накладных расходов на управление виртуальной памятью процессов (доступ к реальному ОЗУ)
- Недостатки:
  - Главным недостатком таких систем является отсутствие многопроцессности:
    - Приложение монолитно и изменяется целиком при обновлении
    - Нет защиты от аварии – «падает» приложение целиком

Архитектура ОС РВ – микроядерная – 1/4

- Микроядерная («клиент-сервер»):
  - Сервисы ОС выносятся в виде серверов на уровень пользователя
  - Микроядро ОС транслирует запросы от СПО (клиентов) к системным сервисам (серверам)
- Основные функции микроядра:
  - Базовые методы IPC (не все) – основная задача микроядра
  - Диспетчер
    - Например, в QNX микроядро управляет потоками, но не процессами
  - Управление виртуальной памятью (не всегда)

Архитектура ОС РВ – микроядерная – 2/4



Архитектура ОС РВ – микроядерная – 3/4

- Преимущества микроядерной архитектуры:
  - Надёжность → каждый сервис отдельное приложение, его легче отладить
  - Гибкость → ненужные сервисы можно исключить без изменения микроядра
  - Отказоустойчивость → «зависший» сервис находится на уровне приложений и может быть перезапущен без перезагрузки системы (но все равно авария!)
  - Расширяемость
  - Легко перейти к распределенной ОС

Архитектура ОС РВ – микроядерная – 4/4

- Недостатки микроядра:
  - Каждый вызов приводит к 4 переключениям режима:
    - Пользовательский клиента → микроядро
    - Микроядро → пользовательский сервера
    - Пользовательский сервера → микроядро
    - Микроядро → пользовательский клиента
  - Например, в Windows NT при переходе к четвертой версии «загнали» значительную часть серверов обратно в ядро, что радикально повысило производительность

Основные ОС РВ: QNX

- Микроядерная
- Относится к Self-hosted
- Портирована на все основные платформы, включая специфические отечественные (например, Эльбрус)
- Доступна в РФ как
  - ЗОСРВ КПДА.00002-01 (двоичная совместимость с QNX 4.25)
  - ЗОСРВ «Нейтрино» КПДА.10964-01

Основные ОС РВ: VxWorks

- Микроядерная
- Относится к Host/Target
- Поддерживает многопроцессорность:
  - Асимметричную (ASMP – Asymmetrical MultiProcessing)
    - Каждый микропроцессор (ядро) исполняет отдельный экземпляр ОС
    - Распределение процессов по процессорам выполняет разработчик СПО

    → Сложно, но есть детерминированностью
  - Симметричную (SMP – Symmetrical MultiProcessing)
    - Многопроцессорная система «видна» как виртуальная однопроцессорная
    - Нагрузка между процессорами распределяется автоматически

    → Упрощает разработку ПО, но нет предсказуемости исполнения