Операционные системы
Лекция 3
Планирование процессов

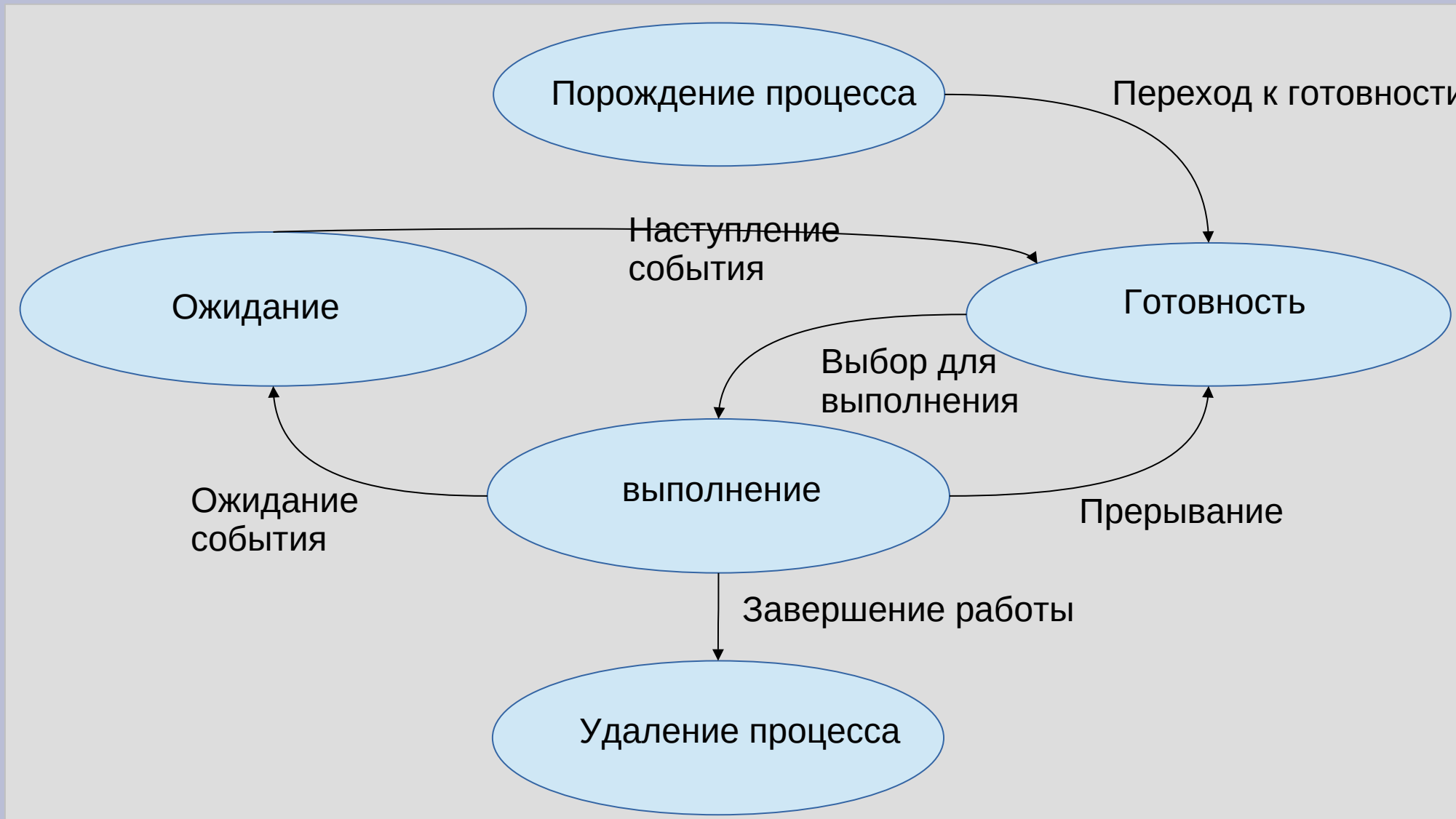
Боровцов Е.Г., 2016 г.

Необходимость планирования

Необходимость планирования выполнения процессов проистекает из необходимости эффективного управления ограниченным набором ресурсов и обеспечения «комфортных» условий решения той или иной задачи

Задачи планирования возникли в связи с появлением первых систем с режимом мультипрограммной работы

Состояния процесса



Моменты планирования

1. После порождения нового процесса и его перехода в состояние готовности
2. После перехода процесса из состояния выполнения в состояние готовности
3. После перехода процесса из состояния выполнения в состояние ожидания
4. После перехода процесса из состояния ожидания в состояние готовности
5. После перехода очередного процесса из состояния выполнения в состояние завершения

Категории алгоритмов планирования

При рассмотрении алгоритмов планирования следует различать три операционных среды:

- **пакетной обработки;**
- **интерактивную;**
- **среду реального времени;**

Кроме того, различается долгосрочное и краткосрочное планирование(диспетчеризация), а также вытесняющее и невытесняющее планирование.

Задачи алгоритмов планирования-1

Для всех систем:

- равнодоступность — предоставление каждому процессу справедливой доли времени центрального процессора;
- принуждение к определенной политике — наблюдение за выполнением установленной политики;
- баланс — поддержка загруженности всех составных частей системы.

Пакетные системы:

- производительность — выполнение максимального количества заданий в час;
- оборотное время — минимизация времени между представлением задачи и ее завершением;
- использование центрального процессора — поддержка постоянной загруженности процессора.

Задачи алгоритмов планирования-2

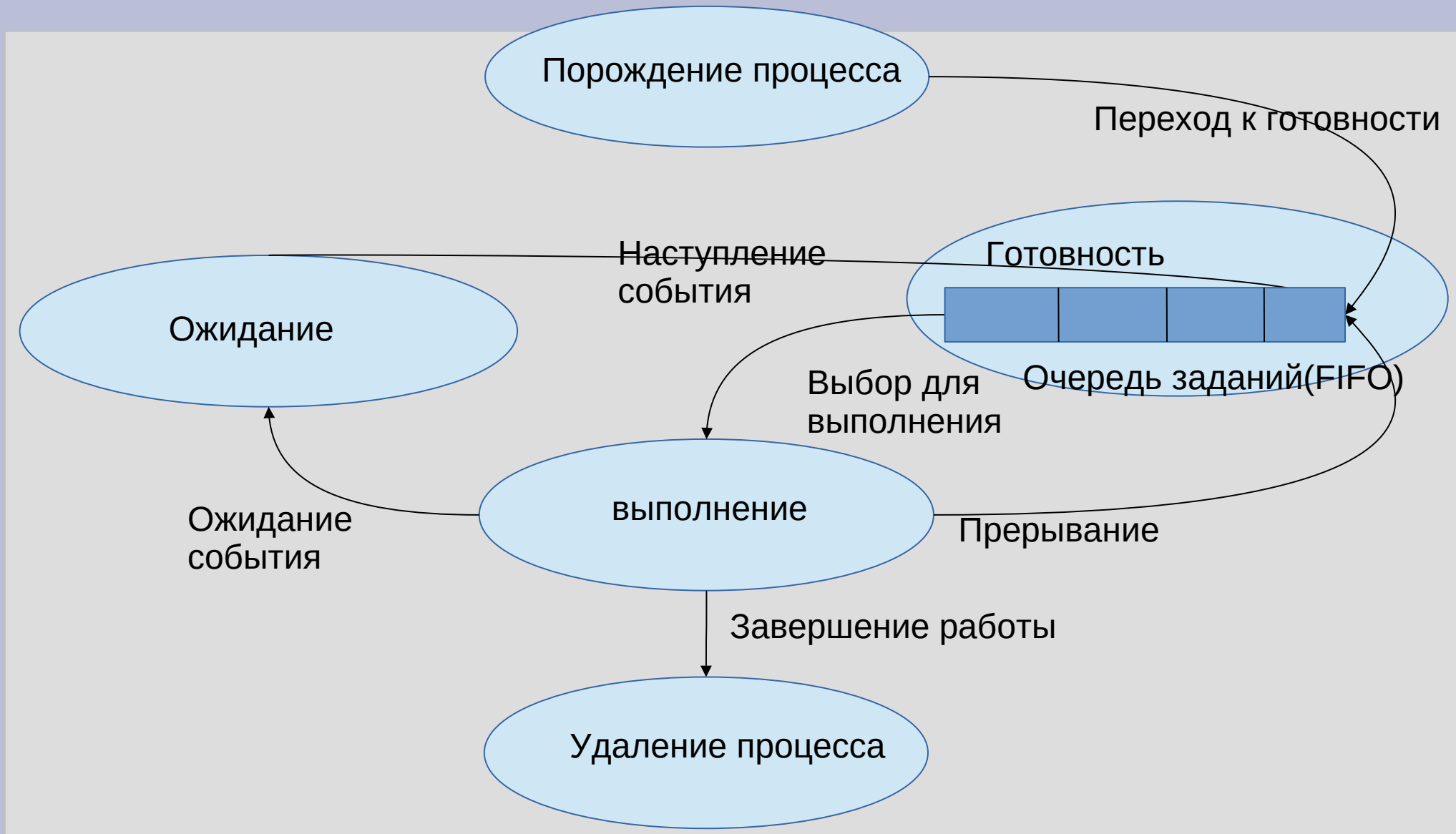
Интерактивные системы:

- время отклика — быстрый ответ на запросы;
- пропорциональность — оправдание пользовательских надежд.

Системы реального времени:

- соблюдение предельных сроков — предотвращение потери данных;
- предсказуемость — предотвращение ухудшения качества в мультимедийных системах.

Алгоритмы планирования: FCFS-First Comes First Served



Алгоритмы планирования: FCFS-First Comes First Served

Достоинства и недостатки алгоритма FCFS:

Сильной стороной этого алгоритма является простота его понимания и такая же простота его программирования. При использовании этого алгоритма отслеживание готовых процессов осуществляется с помощью единого связанного списка. Выбор следующего выполняемого процесса сводится к извлечению одного процесса из начала очереди. Добавление нового задания или разблокированного процесса сводится к присоединению его к концу очереди.

Слабостью алгоритма является возможность монополизации времени одним из процессов, требующего больших затрат процессорного времени. И наоборот, неэффективное использование ресурсов процессора при использовании в рамках одной очереди процессов с сильно различающимися требованиями к процессорному времени и операциям ввода/вывода.

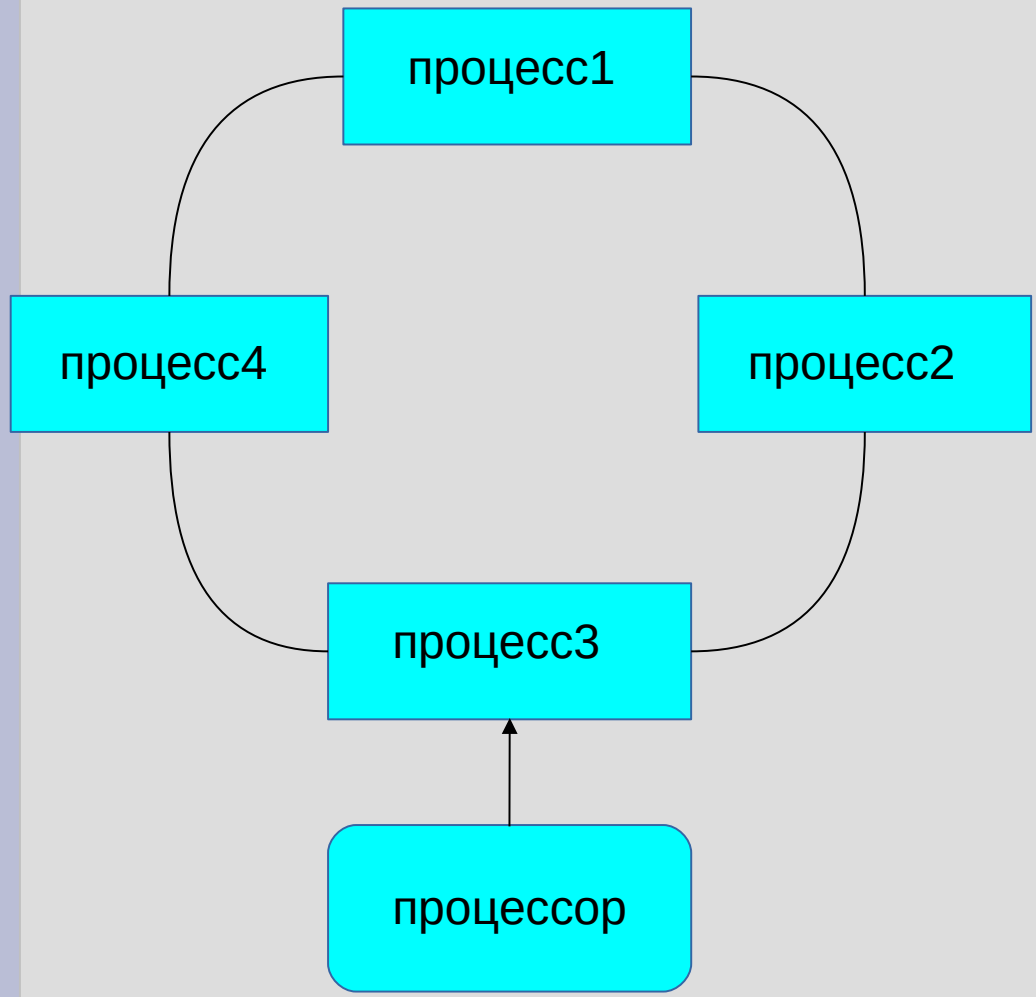
Алгоритмы планирования: JSF- Job Shortest First

Когда в ожидании запуска во входящей очереди находится несколько равнозначных по важности заданий, планировщик выбирает сначала самое короткое задание.

Алгоритм, основанный на выполнении первым самого короткого задания, оптимален только в том случае, если все задания доступны одновременно.

Приоритетной версией алгоритма выполнения первым самого короткого задания является алгоритм первоочередного выполнения задания с наименьшим оставшимся временем выполнения.

Алгоритмы планирования: Циклическое планирование — Алгоритм Round Robin (RR)



Каждому процессу назначается определенный интервал времени, называемый его квантом, в течение которого ему предоставляется возможность выполнения. Если процесс к завершению кванта вре-мени все еще выполняется, то ресурс центрального процессора у него отбирается и передается другому процессу. Если процесс переходит в заблокированное состояние или завершает свою работу до истечения кванта времени, то переключение центрального процессора на другой процесс происходит именно в этот момент.

Алгоритм имеет простую реализацию. Эффективность существенно зависит от размера кванта времени.

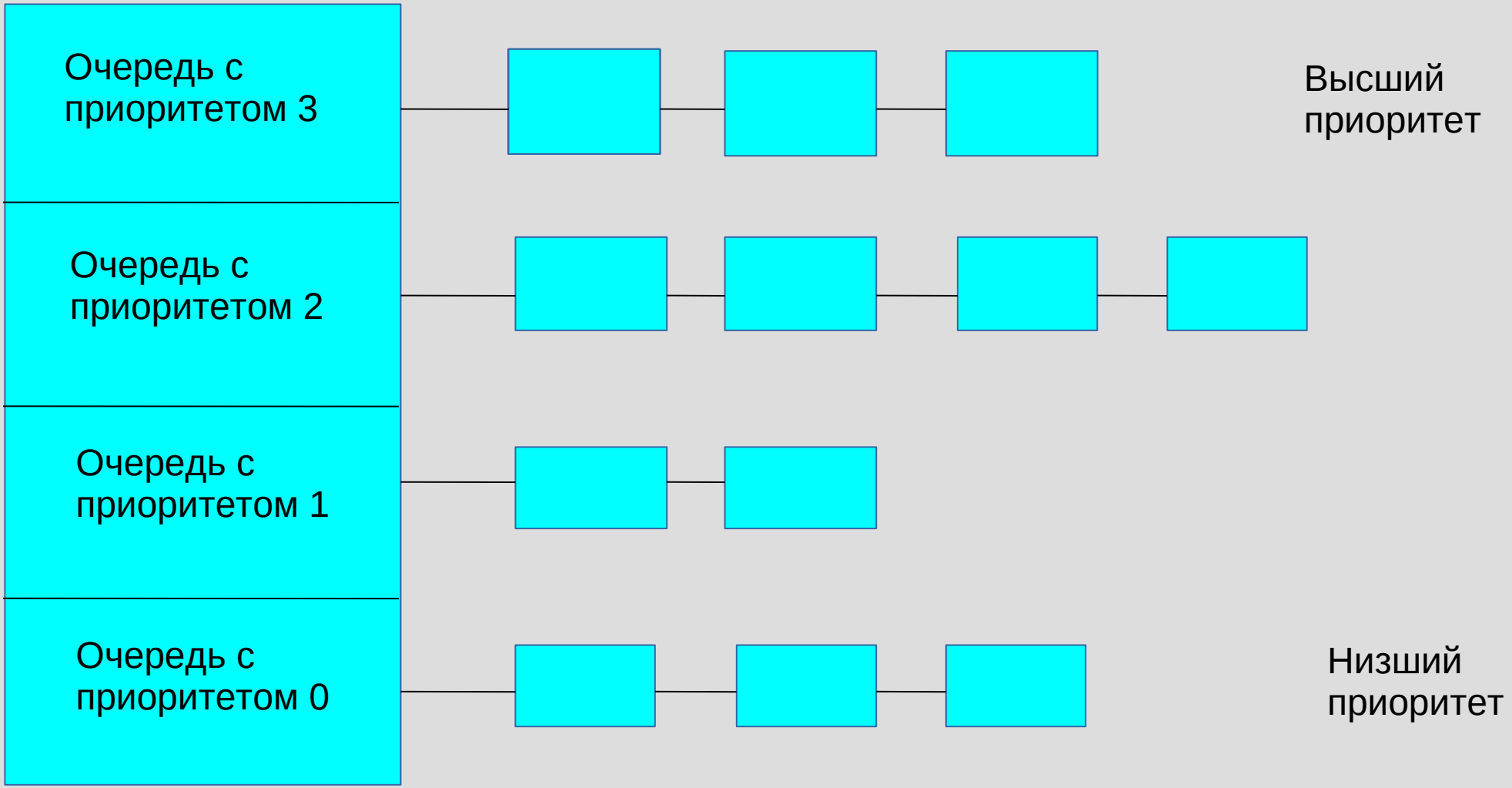
Приоритетное планирование

Каждому процессу присваивается значение приоритетности и запускается тот процесс, который находится в состоянии готовности и имеет наивысший приоритет.

Чтобы предотвратить бесконечное выполнение высокоприоритетных процессов, планировщик должен понижать уровень приоритета текущего выполняемого процесса с каждым сигналом таймера (то есть с каждым его прерыванием). Если это действие приведет к тому, что его приоритет упадет ниже приоритета следующего по этому показателю процесса, произойдет переключение процессов. Возможна и другая альтернатива: каждому процессу может быть выделен максимальный квант допустимого времени выполнения. Когда квант времени будет исчерпан, шанс запуска будет предоставлен другому процессу, имеющему наивысший приоритет.

Приоритеты могут присваиваться процессам в статическом или в динамическом режиме (система «поощрений и штрафов»).

Приоритетное планирование с очередями



Гарантированное планирование

Если в процессе работы в системе зарегистрированы n пользователей, то каждый получит 1/n от мощности центрального процессора. Аналогично этому в однопользовательской системе, имеющей n работающих процессов, при прочих равных условиях каждый из них получит 1/n от общего числа процессорных циклов. Это представляется вполне справедливым решением.

Чтобы выполнить это условие, система должна отслеживать, сколько процессорного времени затрачено на каждый процесс с момента его создания. Затем она вычисляет количество процессорного времени, на которое каждый из них имел право, а именно время с момента его создания, деленное на n. Поскольку также известно и количество времени центрального процессора, уже полученное каждым процессом, нетрудно подсчитать соотношение израсходованного и отпущенного времени центрального процессора.

«Справедливое» планирование

Если пользователь в многопользовательской системе 1 запускает 9 процессов, а пользователь 2 запускает 1 процесс, то при циклическом планировании или при равных приоритетах пользователь 1 получит 90 % процессорного времени, а пользователь 2 — только 10 %.

Чтобы избежать подобной ситуации, некоторые системы перед планированием работы процесса берут в расчет, кто является его владельцем. В этой модели каждому пользователю распределяется некоторая доля процессорного времени и планировщик выбирает процессы, соблюдая это распределение. Таким образом, если каждому из двух пользователей было обещано по 50 % процессорного времени, то они его получат, независимо от количества имеющихся у них процессов.

Особенности планирования в системах реального времени

События, на которые должна реагировать система реального времени, могут быть определены как периодические (происходящие регулярно) или апериодические (происходящие непредсказуемо). Возможно, системе придется реагировать на несколько периодических потоковых событий. В зависимости от времени, необходимого на обработку каждого события, с обработкой всех событий система может даже не справиться. Например, если происходит m периодических событий, событие i возникает с периодом P_i и для обработки каждого события требуется C_i секунд процессорного времени, то поступающая информация может быть обработана только в том случае, если

SUM C(i)/P(i) <=1, i=1...m

Система реального времени, отвечающая этому критерию, называется планируемой. Это означает, что такая система фактически может быть реализована. Процесс, не отвечающий этому тесту, не может быть планируемым, поскольку общее время центрального процессора, требуемое процессу, в совокупности больше того времени, которое этот центральный процессор может предоставить.