УДК 735.29.(32)

# МОДЕЛЬ СОСТОЯНИЙ ВЫЧИСЛИТЕЛЬНОГО ПРОЦЕССА

**Савченко Г.В.**

**Научный руководитель – д-р. техн. наук Легалов А.И.**

***Сибирский федеральный университет***

Каждый параллельный язык программирования опирается на модель вычислений, которая задает особенности параллелизма, определяя управление вычислительными процессами. Существуют различные способы порождения, уничтожения и взаимодействия параллельных процессов, передачи данных между ними. Реализованы технологии параллельного программирования, использующие данные способы, определяющие явное или неявное управление процессами. Существуют различные подходы к описанию управления. Как правило, они не формализованы, выбираются эмпирическим путем в зависимости от прикладной задачи. Предлагается модель состояний, призванная формализовать жизненный цикл процесса программы.

Модель основана на следующих концепциях:

1. Информационных связях между процессами программы. К моменту запуска процесса должны быть предоставлены его аргументы;
2. Зависимости процессов от вычислительных ресурсов, когда к моменту запуска процесса ресурсы, необходимые для его выполнения, были выделены и предоставлены;
3. Наличии субъективных управляющих воздействий, что означает некоторую субъективность запуска операторов при разработке программ, определяющуюся мышлением программиста, а также свободной памятью и другими условиями.

В данной модели (Рисунок 1) процесс извне представлен в виде входов и выходов P = (I, R, C), где

I = (Iin, Iout) служит для передачи входного операционного пакета и выходного информационного пакета соответственно; операционный пакет содержит код операции и входные аргументы. Информационный пакет содержит результаты вычислений. Можно представлять процесс как операцию с уже определенным кодом, тогда на вход Iin подается информационный пакет, содержащий аргументы процесса;

R = (Rin, Rout) служит для передачи входного и выходного ресурсных пакетов. Входной ресурсный пакет состоит из ресурсов, которые назначаются процессу для выполнения, выходной ресурсный пакет состоит из информации о ресурсах, которые были освобождены процессом, или их состояния были изменены;

C = (Cin, Cout) служит для передачи управляющих воздействий, входных и выходных соответственно. Cin принимает команды запуска и остановки процесса, Cout сигнализирует об изменении его состояния.

Основой модели состояния процесса является машина, представляющая собой значительно расширенное понятие конечного автомата в языке моделирования UML (UML State machine, автомат UML). В ней, в отличие от модели синхронного автомата, интервал между входными воздействиями не фиксирован. Машина реагирует на воздействия (внешние события) переходом из состояния в состояние. С каждым состоянием связано входное действие (entry action), выполняющееся, когда машина входит в данное состояние. Каждый переход из исходного в последующее состояние имеет атрибуты:

1. Событие (trigger, триггер), которое приводит к выполнению перехода при условии нахождения в состоянии, принимающем данное событие;
2. Действие перехода (transition action), которое выполняется при переходе между состояниями.
3. Условие перехода (guard condition), определяющее условие, при котором произойдет переход. Конечно, при этом также должен сработать триггер, т.е. произойти событие.

Графическое изображение перехода из начального состояния в состояние «Состояние» изображено на рисунке

.

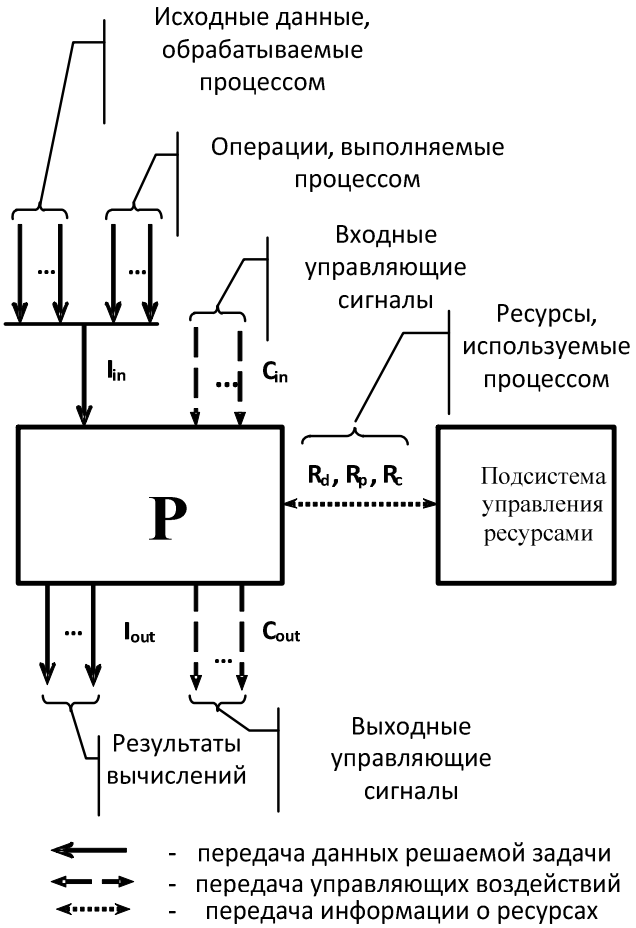



Рисунок 2. Графическое представление вычислительного процесса

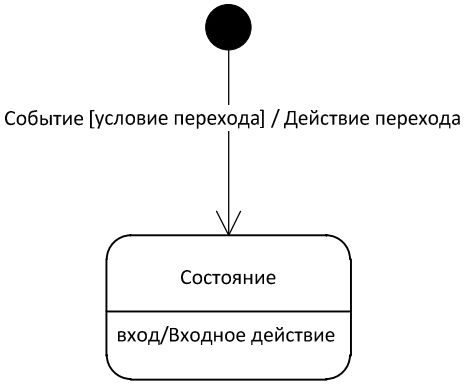



Рисунок 3. Графическое обозначение перехода машины состояний

Машина может находиться в состоянии любое время. Время перехода из состояния в состояние также может быть любым. Действие перехода и входное действие могут выполняться произвольное время. Машина реагирует на события, только находясь в состоянии. Если во время получения события происходит переход между состояниями, событие помещается в очередь. События очереди далее обрабатываются согласно времени прихода. Считается, что одновременно два события происходить не могут.

Рисунок 4 содержит графическое изображение машины состояний процесса в нотации UML. Композитное (составное) состояние «Подготовка выполнения» изображено на рисунке 5.

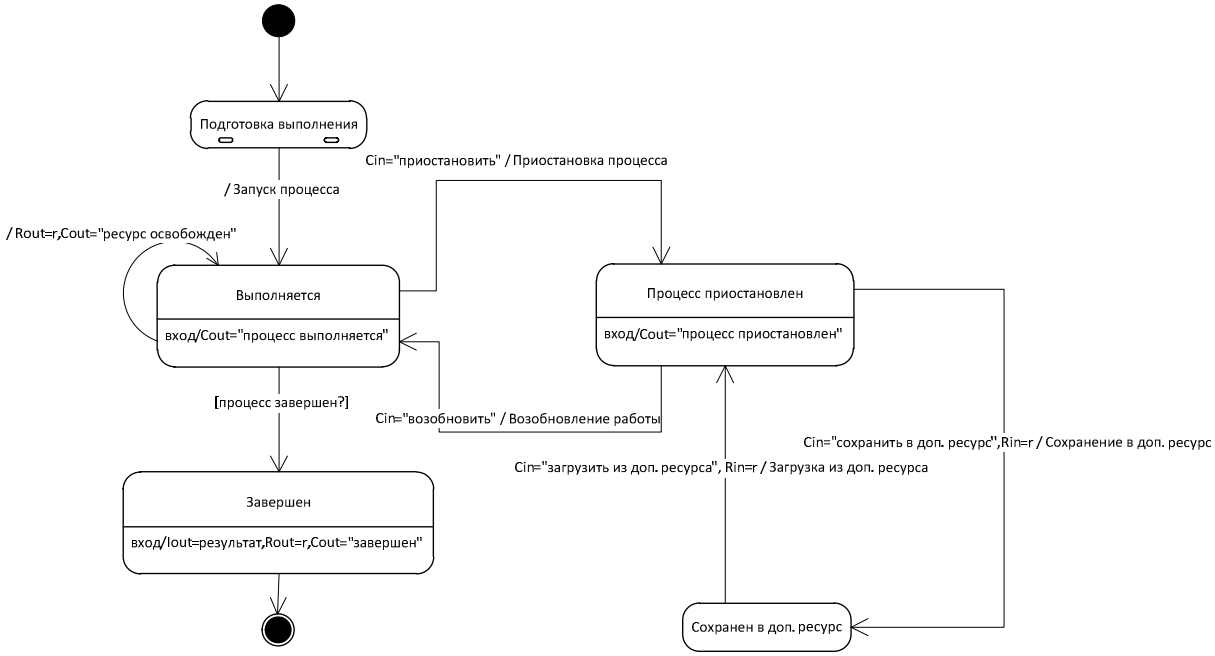
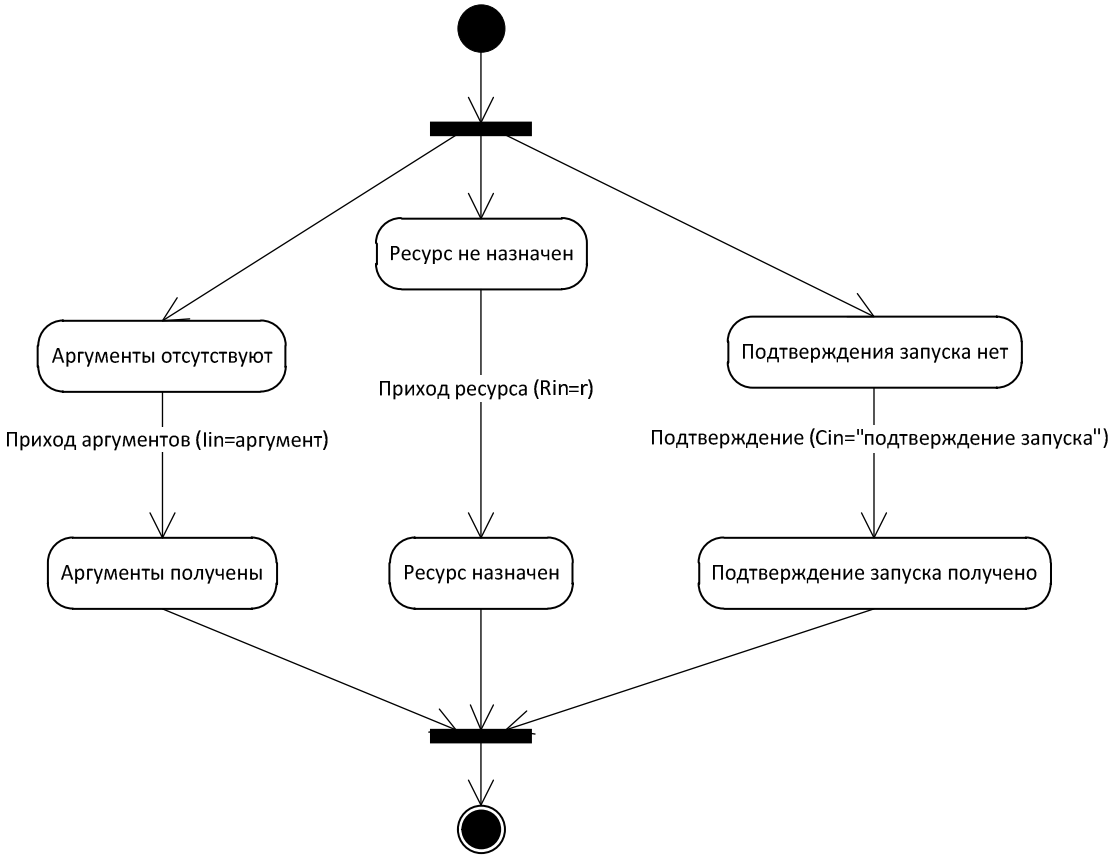



**Рисунок 4. Модель состояний процесса**

**Рисунок 5. Композитное состояние процесса "подготовка выполнения"**

Из описания жизненного цикла процесса следует, что набор состояний, который он проходит от начального до конечного, всегда один и тот же, если не считать возможные приостановки, возобновления, сохранения и загрузки. Отличие состоит в способах достижения состояний «Аргументы получены» (I, Information), «Ресурс назначен» (R, resource), «подтверждение запуска получено» (C, confirmation), являющихся условиями готовности процесса. Состояния достигаются параллельно, как показано на рисунке 5.




Каждое из трех условий готовности I, R, C в сети взаимодействующих вычислительных процессов, в виде которой представляется вычислительная система, может выполняться одним из трех способов:

1. Явно (обозначается как S, Subjective) с точки зрения внешней по отношению к процессу среды подачей управляющих сигналов на входы Iin, Rin, Cin. Рисунок 5 изображает состояния, когда для выполнений трех условий готовности процесса происходит ожидание сигналов (событий).
2. Неявно с точки зрения внешней среды процесса:

a. Процесс может иметь такую внутреннюю организацию, что выполняет проверку условия готовности своими внутренними механизмами. В модели состояний это означает наличие действия перехода, в котором осуществляется удовлетворение условия, например, поиск ресурса. Такое управление называется автоматическим (Automatic, A). Рисунок 6 изображает автоматическое удовлетворение условия готовности ресурсов R, что обозначается наличием действия перехода, в котором осуществляется поиск ресурсов для выполнения. Внешняя среда в данном случае не предоставляет эти ресурсы.

b. Пустое управление (Empty, E), характеризующееся отсутствием проверки условия. Корректность работы взаимодействующих процессов в данном случае обеспечивается их соответствующей организацией. На рисунке 6 условие C удовлетворяется пустым образом, что соответствует безусловному переходу. Такая ситуация возможна, когда каждый процесс обладает достаточным объемом внутренней памяти и используется только один раз, поэтому проверка подтверждений от процессов-приемников результата в данном случае не требуется, также как и проверка других условий запуска.

Всего существует $3^3$=27 способов удовлетворения условий готовности, которые могут быть использованы для классификации языков программирования и вычислительных систем (ВС). В реальных ВС различные способы могут комбинироваться на различных уровнях.

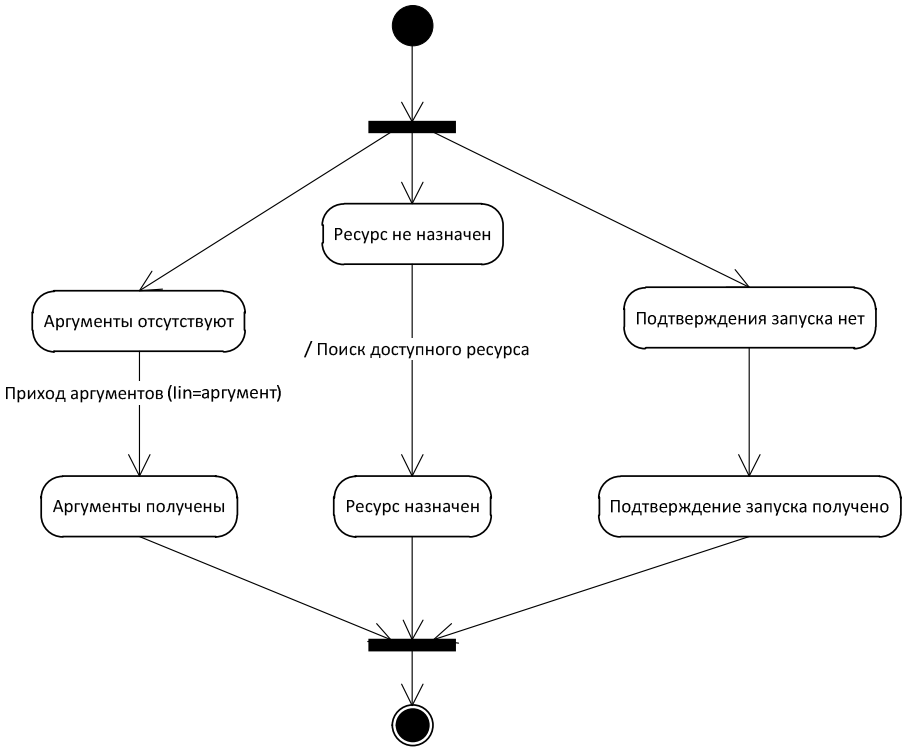



Рисунок 6. Различные способы удовлетворения условий готовности процесса в модели состояний