Лекция №6
Раздел 2. Модельная динамика систем с дискретными событиями

Существует несколько концепций дискретно-событийного моделирования, определяющих различные подходы к декомпозиции динамических процессов, протекающих в моделируемой системе.

1. Концепция событий.
2. Концепция состояний.
3. Концепция параллельных процессов.

3 Концепция состояний в дискретно-событийном моделировании

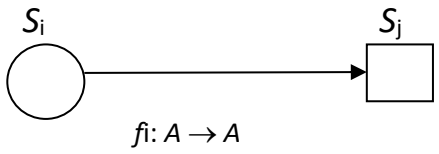
3.1 Понятие схемы состояний

Статической основой схемы состояний является ориентированный и размеченный граф переходов системы из одного состояния в другое. Использование подобного графа для имитации динамических процессов требует разработки алгоритма пересчета системного времени, адаптированного к схеме состояний.

Концепция состояний основана на предположении о том, что в каждый момент системного времени модель находится в одном из состояний множества $S=\{s_1,s_2\ldots,s_n\}$.

Любое текущее состояние системы характеризуется множеством атрибутов $A=\{a_1,a_2\ldots,a_m\}$, которые модифицируются в процессе моделирования.

В состоянии s_i выполняются действия f_i, в результате выполнения которых изменяется множество атрибутов, а система переходит из состояния s_i, в котором она находится, в состояние s_j (Рисунок 10).



Модуль1

Рисунок 10 – Отображение *f* действий, выполняемых системой в состоянии *Si*

Переходы системы из состояния в состояние связаны с возникновением событий. С каждым состоянием s_i с помощью **графа переходов** $\boldsymbol{G_S(S,E)}$ связывается определенное подмножество множества событий $E_i \subset E = \{e_1, e_2, \ldots, e_k\}$, где k – тип события. Одно и то же событие может определять возможности различных переходов. Среди всех элементов множества событий E можно выделить активные события, наступление которых запланировано в будущем, и пассивные события, не запланированные на определенный момент времени, наступление которых зависит от дополнительных условий. Для этих двух категорий событий справедливо:

$$\forall t[\left(\cup_{i=1}^{k} E_i^{act} = E^{act}\right)\&\left(\cup_{i=1}^{k} E_i^{\text{раз}} = E^{\text{раз}}\right)\&(E^{act} \cup E^{\text{раз}}) = E]$$

Алгоритм пересчета системного времени приобретает смысл генерации элементов множества активных событий во времени и их уничтожения. Планирование события типа k, т.е. перевод его в активное состояние, формально определяется, как генерация метки события (e_k,t), принадлежащей множеству, образованному декартовым произведением множества событий и множества моментов времени: $E \times T = \{(e_k, t): (e_k \epsilon E) \& (t \epsilon T)\}$, причем моменты времени определяются на интервале от текущего значения системного времени до верхней границы моделирования $T=\{t: (STIME <= t <= TSIM)\}$.

Планирование событий связано с использованием множества P логических условий или предикатов, определенных на множестве атрибутов.

$P = \{p_1, p_2, \ldots, p_l\}$,

$p_l = p_l(a_1, a_2, \ldots, a_m)$, $l=1..L$.

Если выражение p_l – истина, разрешается планирование определенного подмножества E_i множества событий E на момент времени t в будущем, причем алгоритм образования метки может иметь как детерминированный, так и стохастический характер. Таким образом, предикат выделяет планируемые элементы множества событий E.

Взаимосвязи между предикатами, событиями, определяемыми с их помощью, и алгоритмами образования меток определяются с помощью **графа динамики моделируемой системы $G_T(E, TAU, P)$**, где элементы множеств E и TAU образуют вершины, а элементы множества P – ребра (Рисунок 11).

TAU – множество алгоритмов генерации моментов времени t в информационном потоке, входящих в метки событий (e_k, t), на основе моделирования реализаций случайных величин с заданными законами распределения вероятностей, определяющих интервалы времени между событиями в хронологических потоках.

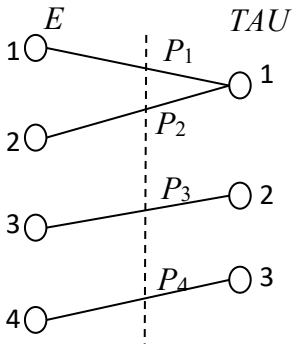


Рисунок 11 – Граф описания динамики моделируемой системы

Таким образом, согласно схеме состояний, моделируемая система описывается двумя графами (переходов и динамики), множеством действий, множеством предикатов и множеством алгоритмов генерации моментов времени наступления событий. Событийная основа выполняет вспомогательные функции идентификации возможного перехода. При этом моделируемую систему следует рассматривать, как **конечный автомат, функционирующий во времени**.

3.2 Автоматно-событийная модель

Автоматно-событийная модель показана на Рисунке 11.

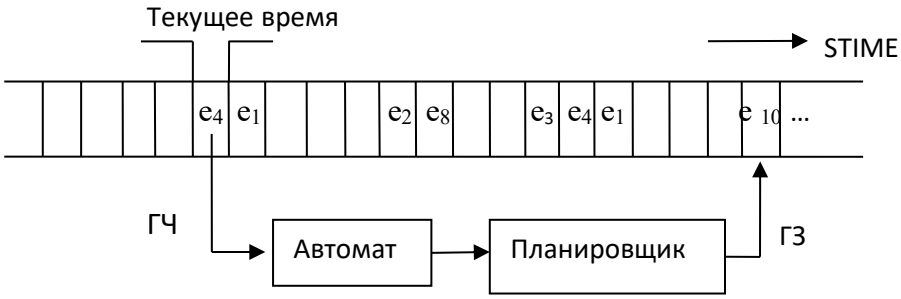
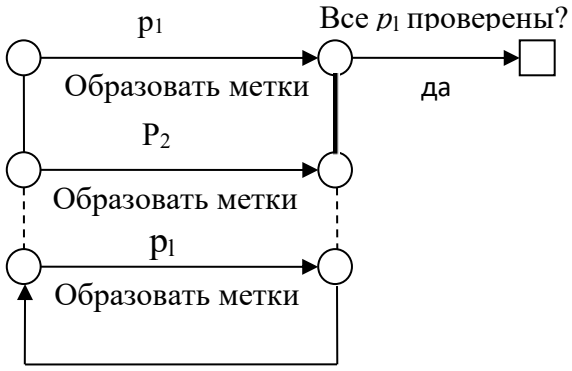


Рисунок 11 – автоматно-событийная модель

Алгоритм функционирования автоматно-событийной модели состоит из двух этапов:

1. Алгоритм функционирования автомата.
   1.1. Сдвиг головки чтения (ГЧ) на ближайший правый непустой квант, т.е. пересчет системного времени.
   1.2. Определение нового состояния s_i в соответствии с графом переходов.
   $E \times S \to S$
   1.3. Выполнение действий f_i, соответствующих состоянию s_i.
   1.4. Передача управления планировщику.
2. Алгоритм функционирования планировщика (Рисунок 12).
   2.1. Проверка истинности предикатов, принадлежащих множеству *P*.
   2.2. Выделение элементов множества событий *E*, соответствующих истинным предикатам.
   2.3. Образование меток (e_k,t) и запись их на ленту фреймов с помощью головки записи (ГЗ).
   $$(e_k, t)\epsilon E \times T$$
   2.4. Передача управления автомату.



Модуль 2

Рисунок 12 – Структура планировщика событий

Над стрелками записаны предикаты, под стрелками – операторы планирования событий, причем связи между предикатами, соответствующими типами событий e_k и моментами времени t устанавливаются через граф динамики.

На Рисунке 13 представлена схема алгоритма функционирования монитора моделирования, поддерживающего автоматно-событийную модель. Процесс функционирования такой модели связан с перемещениями по графу переходов, происходящими во времени в соответствии с закономерностями изменения состояния исследуемой системы, определяемыми графом динамики.

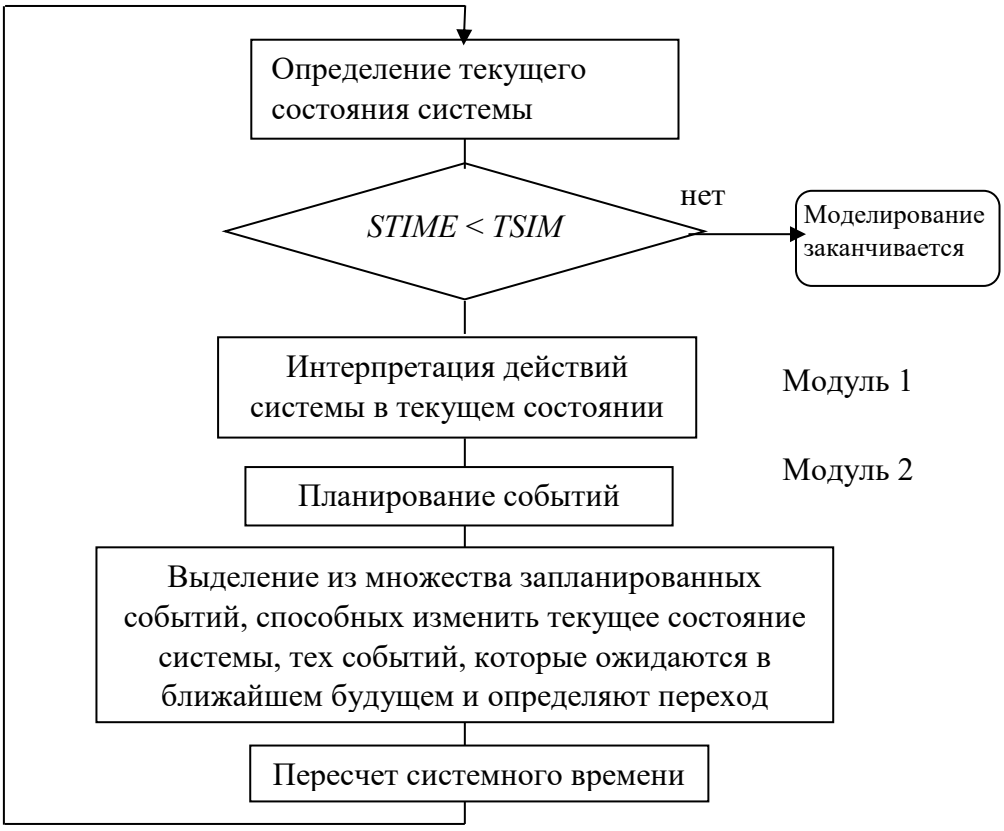


Рисунок 13 – Схема алгоритма функционирования монитора моделирования, поддерживающего автоматно-событийную модель

3.3 Программная реализация автоматно-событийной модели

С содержательной точки зрения программа имитации, основанная на концепции состояний, состоит из двух частей:

1. Декларирующая или описательная часть – определяется двумя графами: переходов ***Gs(S, E)*** и динамики ***G_T(E, TAU, P)***, которые удобно объединить в один граф ***G(S, E, TAU, P)***, называемый **объединенным графом динамики и переходов**.
2. Интерпретирующая часть – состоит из двух компонентов:
   - выполнение действий fi;
   - образование меток (e_k,t).

Действия fi могут быть реализованы в виде процедур, написанных на выбранном базовом языке программирования. Предикаты P, определяющие планирование событий, могут задаваться арифметическим выражением, что означает планирование перехода через заданный этим выражением интервал модельного времени (***безусловный переход***),

или логическим выражением, что означает переход при выполнении условия, заданного этим выражением (***условный переход***). Описание условия перехода может отсутствовать в автоматной части, в этом случае предикаты задаются при описании действий (процессов управления) и определяют ***возможный переход***. *TAU* – это методы генерации информационных потоков.

Автомат и планировщик работают с информационной структурой следующего вида. Это массив дескрипторов состояний автомата, в которые возможен переход из текущего состояния (Рисунок 14). Используется массив, а не список, т.к. количество состояний автомата конечно и не может измениться в процессе моделирования после определения графа переходов и динамики *G*.

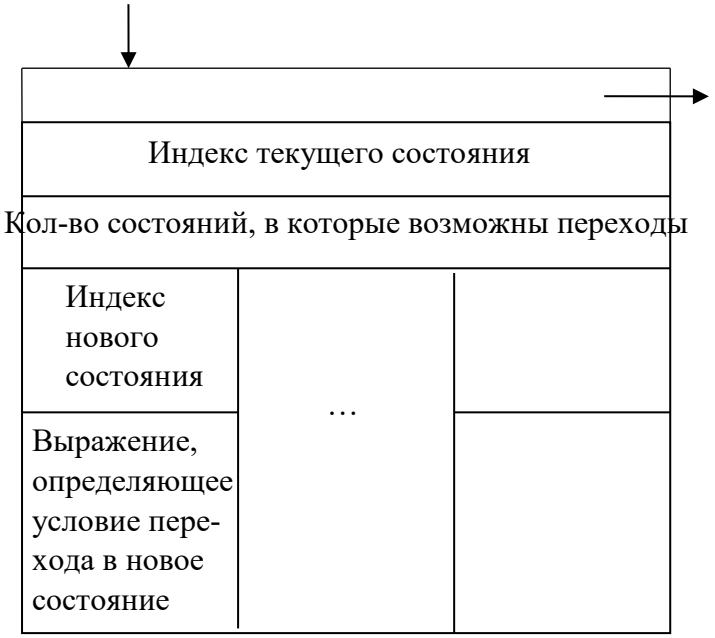


Рисунок 14 – Дескриптор состояний автомата

Каждое состояние автомата, описанное дескриптором, определяется индексом состояния, количеством состояний, в которые возможны переходы из текущего состояния, а также массивом записей о возможных переходах, включая индекс нового состояния и выражение, определяющее условие перехода, которое может быть задано переменной процедурного типа или некоторым заранее определенным идентификатором (например, номером).

В соответствии с двумя частями программы имитации, основанной на автоматно-событийной концепции, программа реализации автоматно-событийного монитора моделирования состоит из двух подсистем:

1. Транслятор, обрабатывающий описание объединенного графа динамики и переходов, определенное в декларирующей части программы.
2. Монитор моделирования, управляющий выполнением действий f_i, заданных в интерпретирующей части программы и реализованных в виде процедур, выполняющих проверку множества предикатов и реализующих алгоритмы генерации меток событий.

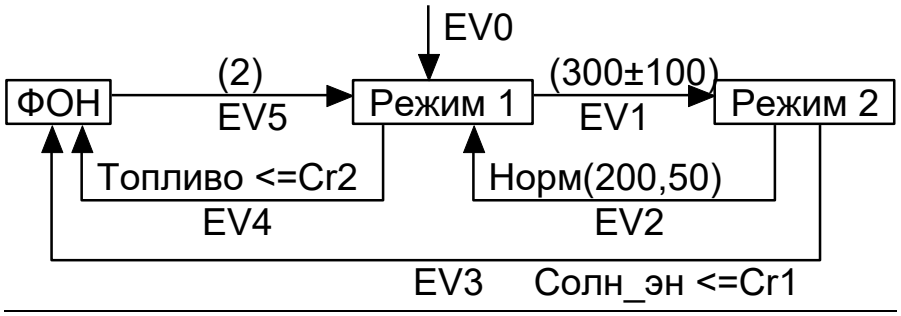
3.4 Пример программы, реализующей автоматно-событийную модель

В качестве примера рассмотрим программу, реализующую конечный автомат, моделирующий управление топливной системой космического аппарата во время полета.

```
Ресурсы     Солн_Эн, Топливо
CONST       Cr1 = ... (* Критический уровень Солн_Эн *)
            Cr2 = ... (* Критический уровень Топливо *)
Автомат
            Фон → Режим_1 / 2;
            Режим_1 → Режим_2 / РАВН(300+-100), Фон;
            Режим_2 → Режим_1 / НОРМ(200,50), Фон;

Процессы управления:
sit_1:  напечатать «Критический уровень Солн_Эн»;
        установить состояние «Фон»;
        активизировать ситуацию Солн_Эн <= Cr1; Возврат;
sit_2:  напечатать «Критический уровень Топливо»; Останов;
Старт: (* инициализация *)
        установить состояние «Режим_1»;
        активизировать sit_1 по условию Солн_Эн<=Cr1;
        активизировать sit_2 по условию Топливо<=Cr2.
```

На Рисунке 15 показан Объединенный граф переходов и динамики для данной модели.



EV0 – «начало»;
EV1 – переход из Режим_1 в Режим_2 (безусловный);
EV2 – переход из Режим_2 в Режим_1 (безусловный);
EV3 – переход из Режим_2 в Фон (условный);
EV4 – переход из Режим_1 в Фон (условный);
EV5 – переход из Фон в Режим_1 (безусловный).

Рисунок 15 – Объединенный граф переходов и динамики