

Способы продвижения времени в имитационной модели

С постоянным шагом

- Непрерывные модели (дифф. уравнения, ОДУ)
- Тактовые модели (разностные уравнения)

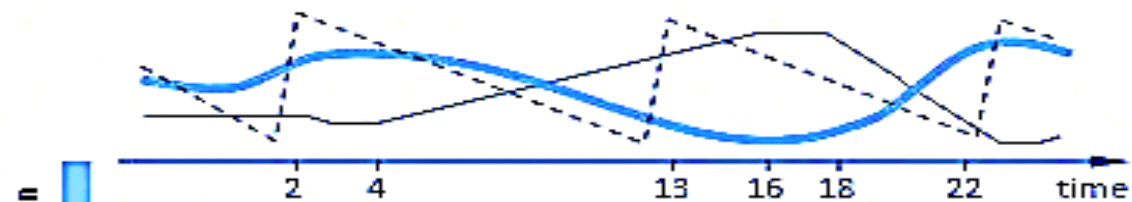
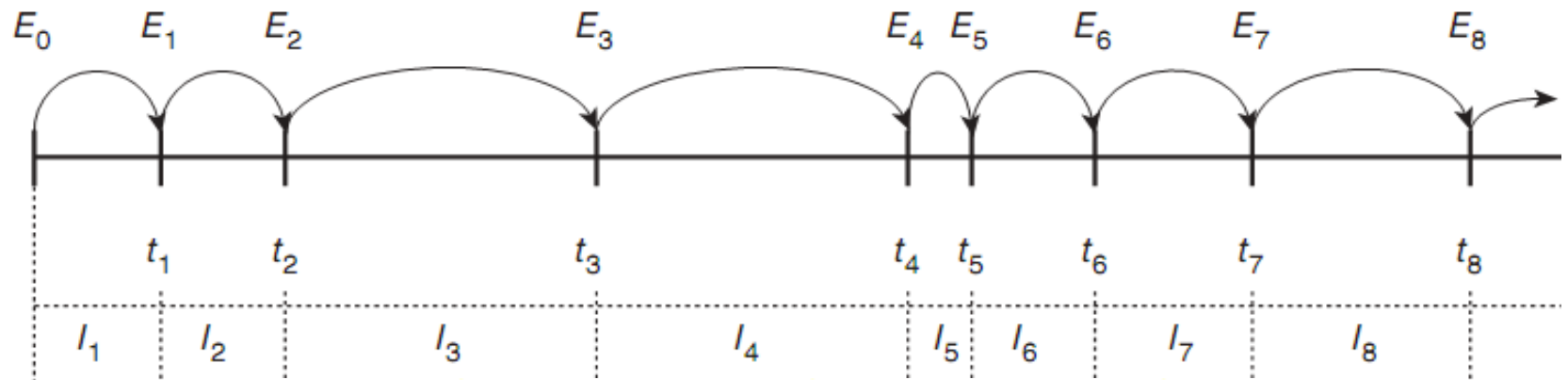
От события к событию

- Дискретно-событийные модели (DES)

Гибридные модели

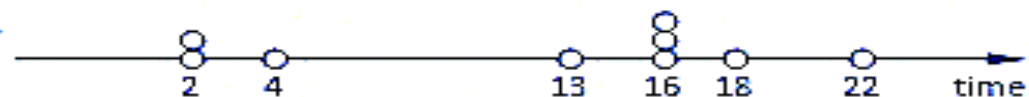
- Совместная работа компонентов разного рода (DES+ОДУ)
- Переключение режимов «непрерывного» компонента

Алгоритм формирования модельного времени DES

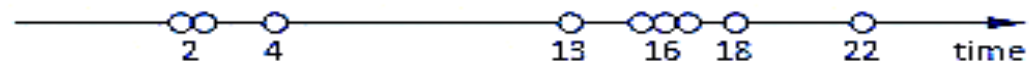


The dynamics of the real world system is continuous

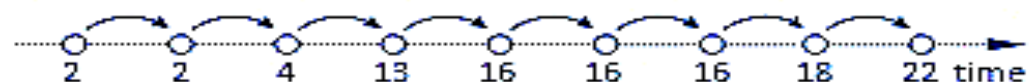
Abstraction
↓



We consider only important moments in the system lifetime – events



Simultaneous events are serialized during the simulation



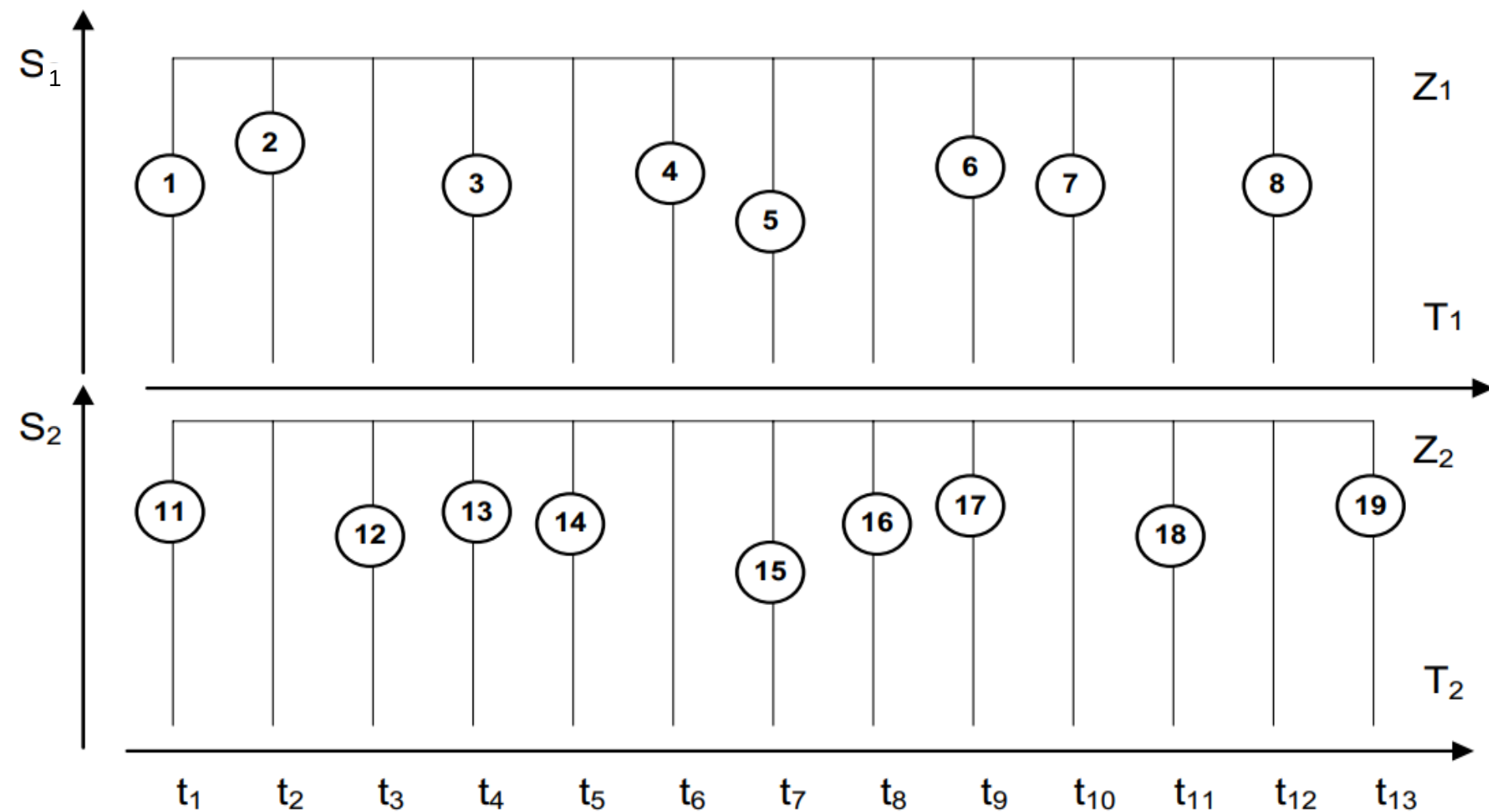
Simulation time is discrete – it jumps from one event to another

ПОСТРОЕНИЕ ИМИТАЦИОННОГО ПРОЦЕССА

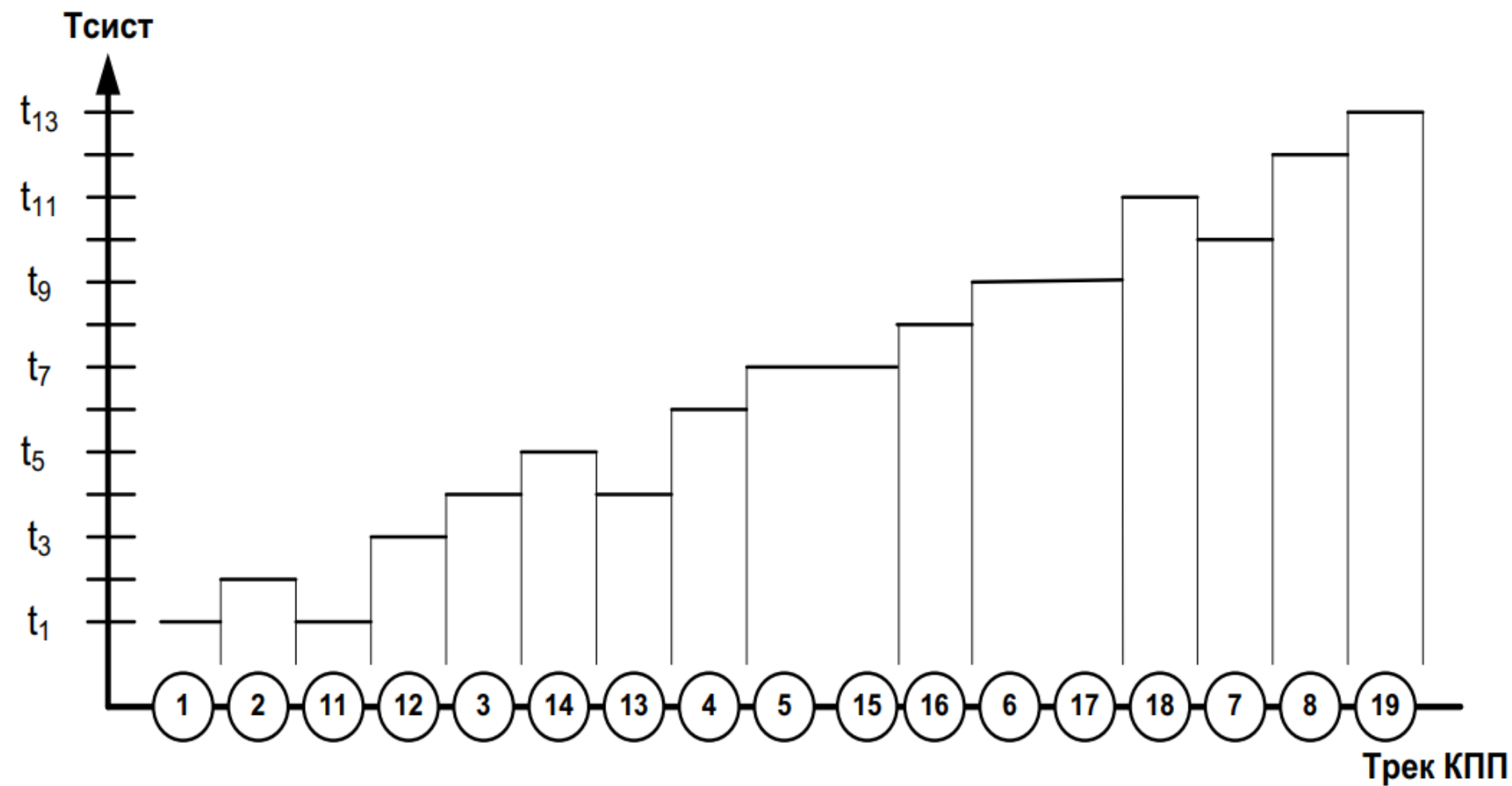
При дискретно-событийном подходе (discrete event system) имитационный процесс рассматривается как один последовательный вычислительный процесс, на который отображается *система параллельных взаимосвязанных процессов*.

Такой последовательный вычислительный процесс называют **квазипараллельный процесс (КПП)**.

Главное требование к квазипараллельному процессу – реализация всех событий на КПП без нарушения их причинно-следственных отношений, существующих в исходной системе процессов.

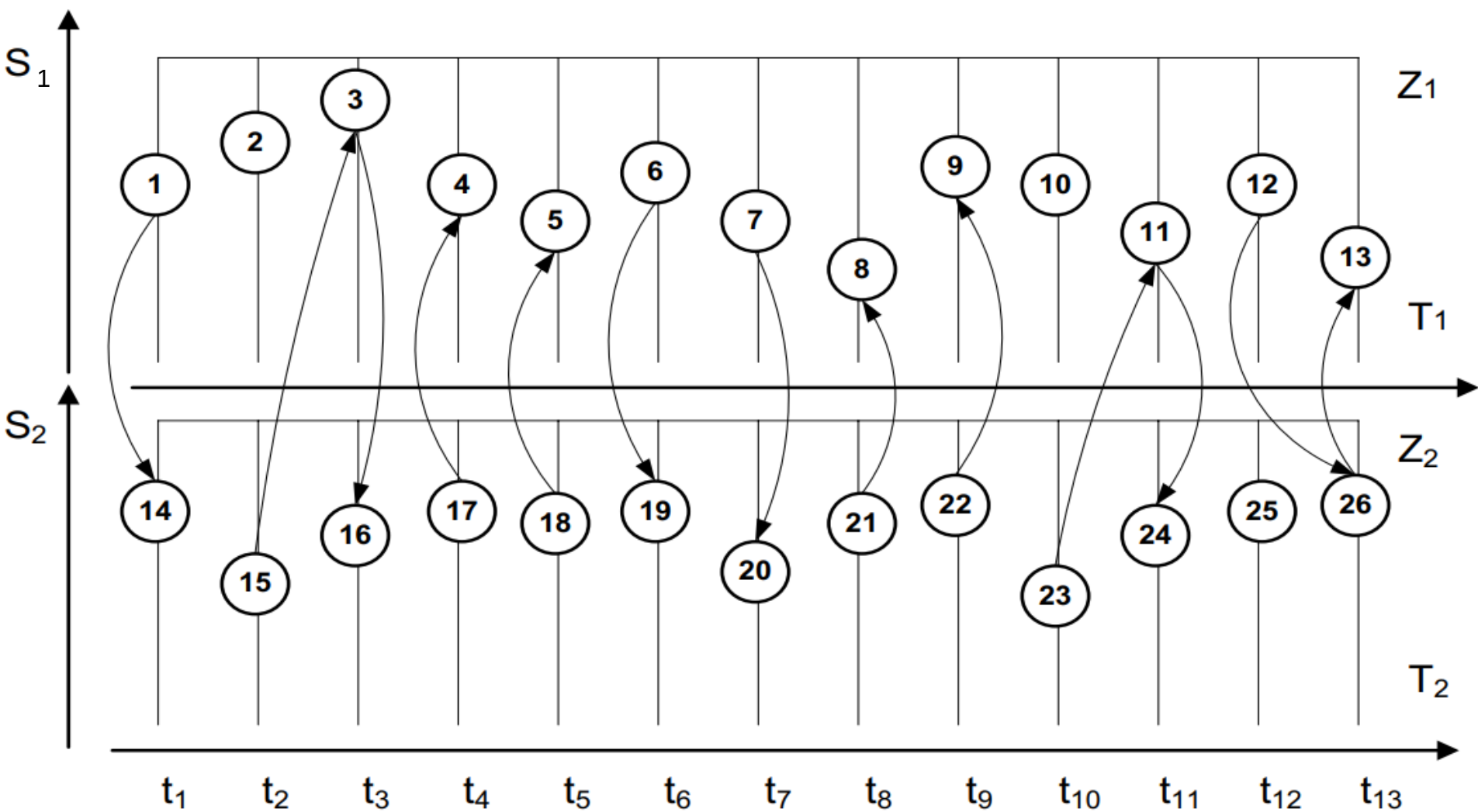


Например, заданы два процесса Z_1 и Z_2



Этот трек КПП – без учета влияния отношения сцепления Z1 и Z2 на последовательность выполнения операторов

процессы Z1 и Z2 с учетом сцепленности операторов

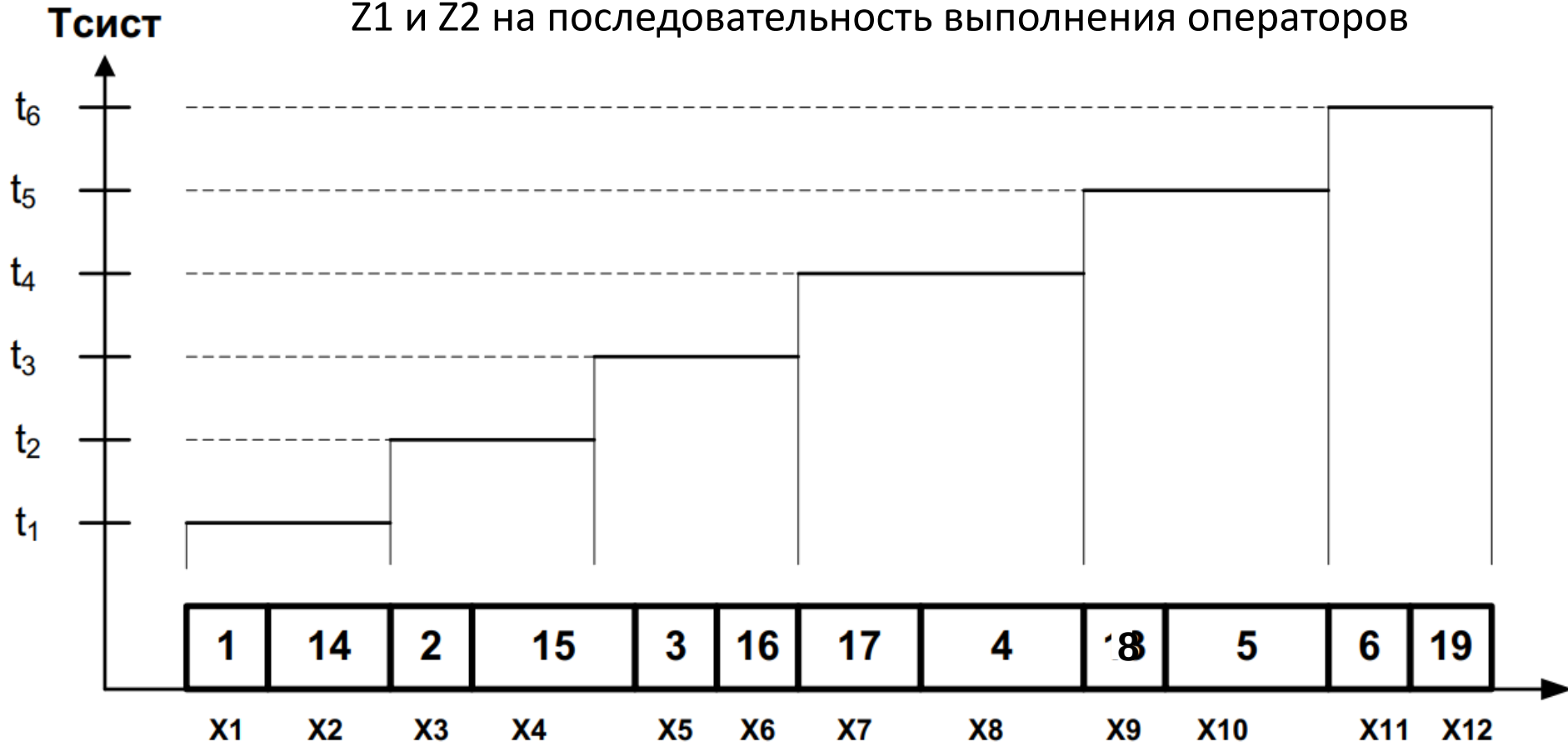


Элементарные операторы процесса Z1 пронумерованы от 1 до 13, а для процесса Z2 - от 14 до 26.

Возможны четыре типовых случая:

- а) моделируется один процесс Z1, причём его операторы не сцеплены между собой;
- б) моделируется процесс Z1 без ограничения на сцепленность элементарных операторов;
- в) моделируются два процесса Z1 и Z2, при этом эти процессы не сцеплены между собой;
- г) моделируются два сцепленных процесса Z1 и Z2 .

Этот трек КПП – с учетом влияния отношения сцепления Z1 и Z2 на последовательность выполнения операторов



возможный порядок вычислений:

***$h_1, h_{14}, (h_2, h_{15}), h_3, h_{16}, h_{17}, h_4, h_{18}, h_5, h_6, h_{19}, h_7, h_{20},$
 $h_{21}, h_8, h_{22}, h_9, (h_{10}, h_{23}), h_{11}, h_{24}, (h_{12}, h_{25}), h_{26}, h_{13}$***

график времени по оси $T_{сист}$ будет монотонно возрастающим.

Алгоритм определения порядка вычисления элементарных операторов в КПП

Пусть заданы треки процессов Z^i ($i = \overline{1, n}$): $\langle \{h_j^i\}, \beta^i \rangle$ для всех i . Пусть текущее значение времени равно t и все элементарные операторы h^i , у которых $t^i \leq t$, вычислены. Для всех h_j^i , имеющих $t^i = t$ и обладающих условным временным оператором $(h_j^i)^t$, определим для каждого очередной момент времени t_{j+1}^i сцепления инициатора по своему треку, задаваемый этим временным оператором. Получим множество $\{t_{j+1}^i\}$. Назовем его *активным временным множеством*.

Это множество содержит по одному значению от каждого процесса, остановленного на элементарном операторе, содержащем временное условие продвижения инициатора. Определим очередное значение времени по формуле:

$$t_0 = \min \{t_{j+1}^i\} \quad (2)$$

Последовательное применение формулы 2 строит упорядоченное множество $T^{\text{КПП}}$. Легко доказать, что множества $T^{\text{КПП}}$ и $(\bigcup_{\forall i} t^i)$ равны.

Выполнение каждого элементарного оператора назовём **событием**.

Событие **активное**, если оно следует в треке за элементарным оператором, содержащем h_t

Событие **пассивное**, если оно следует в треке за элементарным оператором, содержащем h_l

Множество событий, происходящих
в один и тот же момент модельного времени, назовем
классом одновременных событий (КОС)

В каждом КОС содержится только одно активное событие:

- а) все остальные события в КОС, если они есть, являются пассивными;
- б) количество КОС равно мощности объединенного множества времен;
- в) первое событие в любом КОС – активное событие.

Алгоритм формирования КОС

1. Выполняется активное событие;
2. Проверяются условия всех возможных к этому времени пассивных событий;
3. Если выполняется условие пассивного события, то оно вычисляется;
4. Выполнение повторяется с п.2.
5. КОС завершен, если никакие условия, заданные операторами $h_{\mathcal{L}}$ во всех треках, не выполняются.

ОРГАНИЗАЦИЯ ПРОГРАММЫ ДИСКРЕТНО-СОБЫТИЙНОЙ МОДЕЛИ

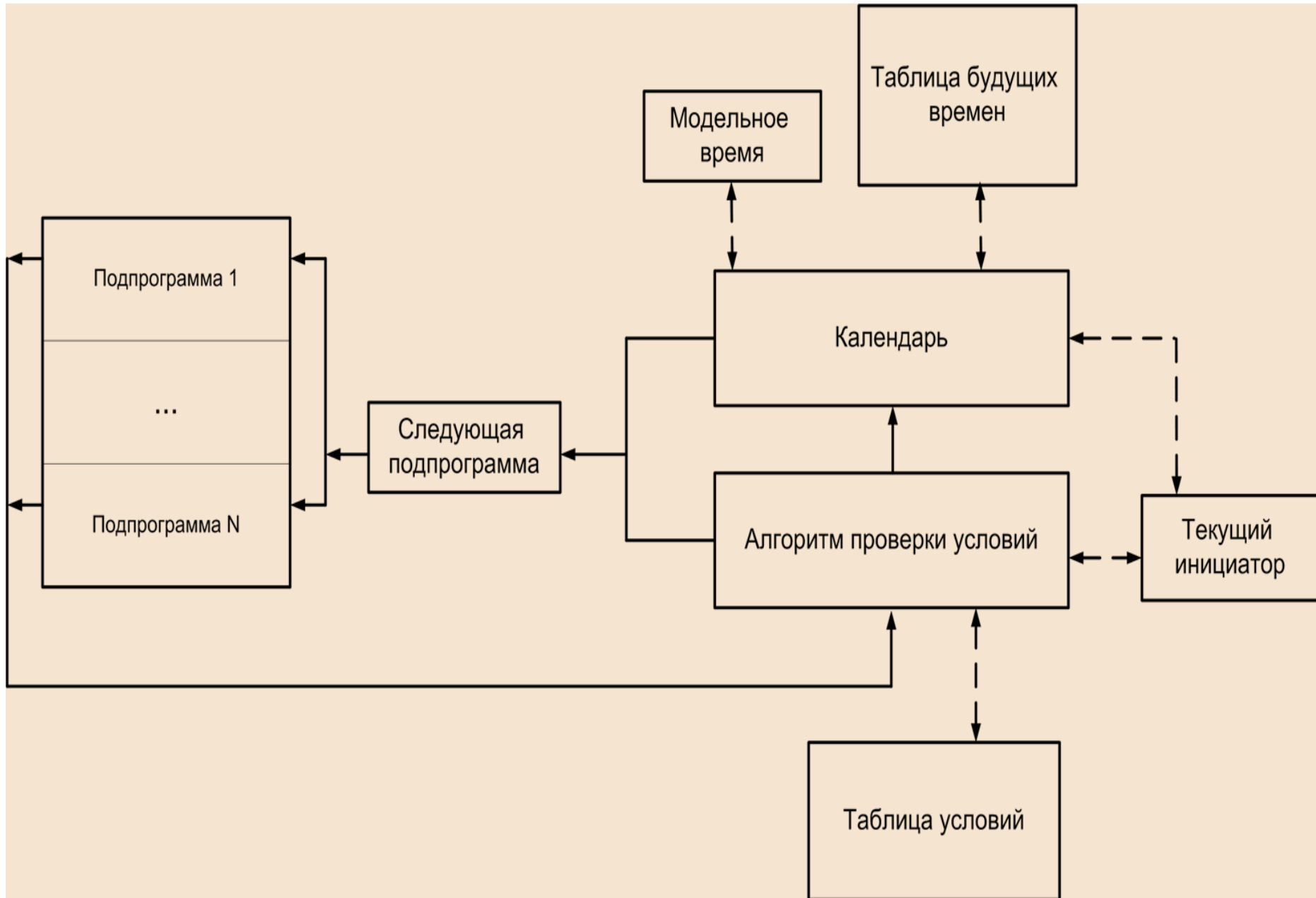
Моделирующий алгоритм

Моделирующий алгоритм *сканирующего* типа

состоит из следующих составных частей:

- подпрограммы событий, реализующие элементарные операторы;
- алгоритм формирования модельного времени;
- алгоритм выбора очередного класса одновременных событий;
- алгоритм генерирования КОС.

Моделирующий алгоритм сканирующего типа



ОРГАНИЗАЦИЯ МОДЕЛИ

Моделирующий алгоритм

Параметр ВРЕМЯ – содержит текущее значение модельного времени;

Параметр ИНИЦИАТОР – содержит значение текущего инициатора (ссылку на локальную среду процесса).

КАЛЕНДАРЬ – алгоритм, реализующий монотонно возрастающее продвижение модельного времени и начало нового КОС в системе.

АПУ – Алгоритм Проверки Условий, обеспечивающий построение КОС для текущего значения модельного времени.

ТБВ – Таблица Будущих Времен, каждая строка ТБВ соответствует одному процессу и содержит элементы описания активного события.

Таблица будущих времен

Время	ID Инициатора	Индекс подпрограммы события	ID События
Время появления активного события и запуска подпрограммы, описывающей соответствующий оператор	Идентификатор динамического объекта, направляемого на выполнение подпрограммы события	Индекс подпрограммы, выполняемой динамическим объектом при появлении описанного активного события.	Идентификатор, определяющий принадлежность простого временного условия к событию, происходящему при выполнении одного простого или нескольких условий (составного условия)

Таблица условий

Условие	ID Инициатора	Индекс подпрограммы события	ID События
Условие появления пассивного события и запуска подпрограммы, описывающей соответствующий оператор	Идентификатор динамического объекта, направляемого на выполнение подпрограммы события	Индекс подпрограммы, выполняемой динамическим объектом при появлении описанного пассивного события	Идентификатор, определяющий принадлежность простого временного условия к событию, происходящему при выполнении одного простого или нескольких условий (составного условия)

Интегрированная модель

Подпрограмма событий 1-го типа	Интегрированная имитационная модель включает в себя набор подпрограмм, каждая из которых является элементарным оператором, выполняемым при наступлении активного или пассивного события.
...	
Подпрограмма событий 1-го типа	

ОРГАНИЗАЦИЯ МОДЕЛИ

МОДЕЛИРУЮЩИЙ АЛГОРИТМ

Подпрограмма события представляет собой программную реализацию одного **элементарного оператора** (оператор состояния, оператор условия продвижения инициатора, навигационный оператор).

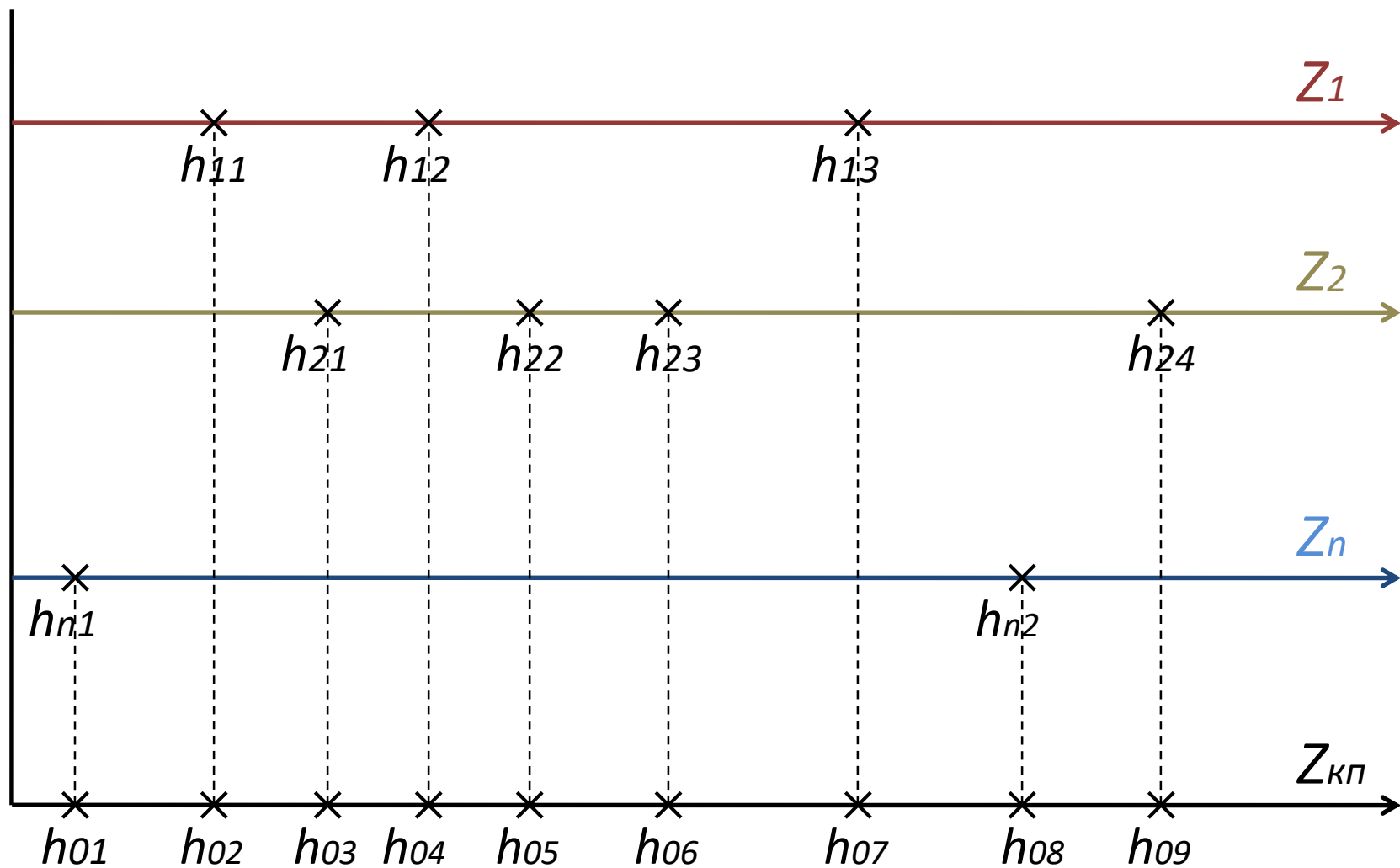
В этих подпрограммах все параметры могут быть **глобальными**.

В большинстве случаев часть параметров может быть *локализована*.

Все параметры, через которые осуществляется обмен, являются глобальными.

Если подпрограмма события реализует **объединенный** элементарный оператор, то она должна иметь доступ к значению инициатора, определяющего *локальную среду* данного процесса.

Подпрограммы событий после своего выполнения передают управление в алгоритм АПУ.



ОРГАНИЗАЦИЯ МОДЕЛИ

МОДЕЛИРУЮЩИЙ АЛГОРИТМ

Алгоритм КАЛЕНДАРЬ определяет первое активное событие в новом КОС в соответствии с

$$t_0 = \min t_{j+1}^i, (\forall i)$$

Алгоритм:

- 1) поиск минимального значения в столбце 1 ТБВ. Пусть это значение равно t_k , где k - номер строки ТБВ
- 2) ВРЕМЯ := t_k
- 3) ИНИЦИАТОР := <значение столбца 2 в ТБВ по строке k >
- 4) C := <значение столбца 3 в ТБВ по строке k >
- 5) Затирание k -ой строки ТБВ
- 6) Передача управления по адресу, хранящемуся в C

Из алгоритма видно, что КАЛЕНДАРЬ ведёт модельное время и иницирует выполнение активного события - первого события в каждом КОС.

ОРГАНИЗАЦИЯ МОДЕЛИ

Моделирующий алгоритм

Алгоритм проверки условий (АПУ) генерирует пассивные события КОС:

- 1) Расчет всех логических условий, заданных в столбце 1 ТУ. В зависимости от результата, полученного при вычислении логического условия j -ой строки ТУ, выполняется шаг 2 либо шаг 3.
- 2) Если логическое условие равно 0 (*ложь*), то $j := j+1$ и повторяется шаг 1.
- 3) Если логическое условие равно 1 (*истина*), то происходит разбор j -й строки:
 - ИНИЦИАТОР := <значение столбца 2 в ТУ по j -й строке>;
 - $D :=$ <значение столбца 3 в ТУ по j -й строке>;
 - затирание j -й строки ТУ;
 - передача управления по адресу, хранящемуся в D .
- 4) Если все логические условия в столбце 1 равны 0 и нет ни одного условия, равного 1, управление передаётся программе КАЛЕНДАРЬ, так как исчерпаны все события текущего КОС и необходим переход к новому КОС, начинающемуся с активного события.

ОРГАНИЗАЦИЯ МОДЕЛИ

Моделирующий алгоритм

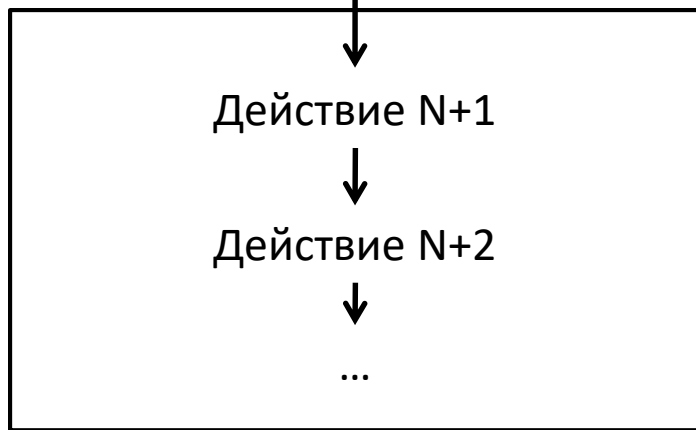
Каждая подпрограмма hi в ходе своего выполнения меняет состояние системы и определяет условие продвижения инициатора своего процесса по треку.

Если подпрограмма hi содержит условие типа ht , то ht заполняет строку в ТБВ, определяет момент активизации инициатора.

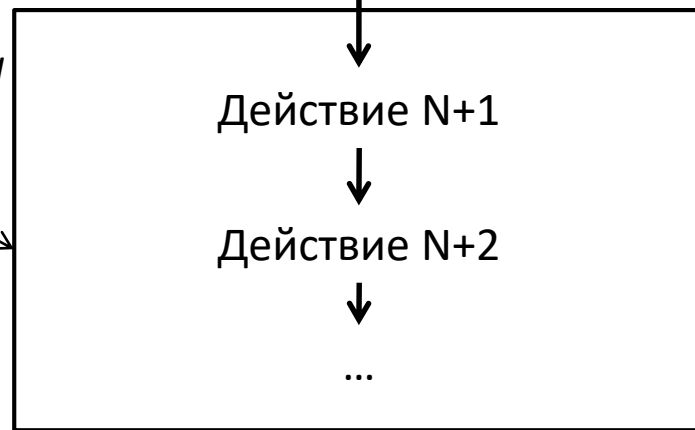
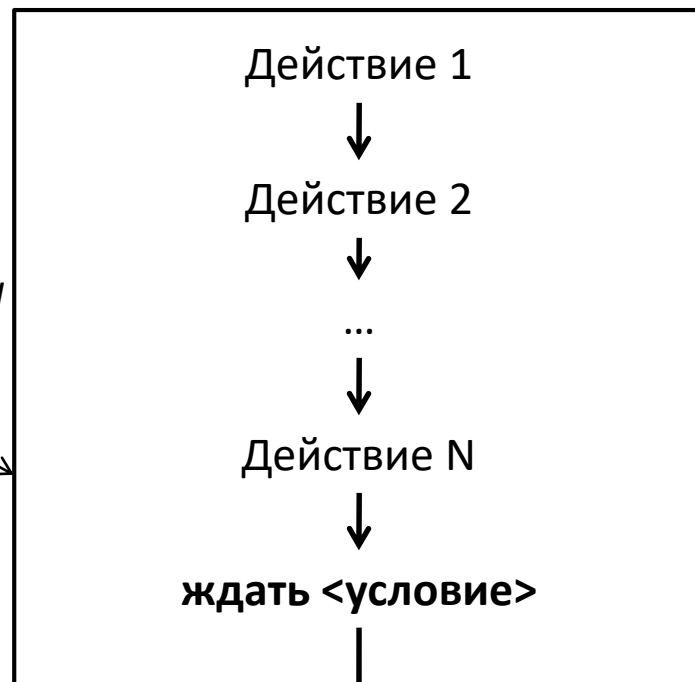
Навигационный оператор hn в составе ht определяет адрес следующей по треку подпрограммы событий. В эту же таблицу помещается и значение текущего инициатора (ссылка на локальную среду).

Если подпрограмма hi содержит условие типа hl , то hl заносит строку в ТУ, помещая в столбец 1 логическое условие, а в остальные - инициатор и адрес следующей по треку подпрограммы событий.

Активное событие



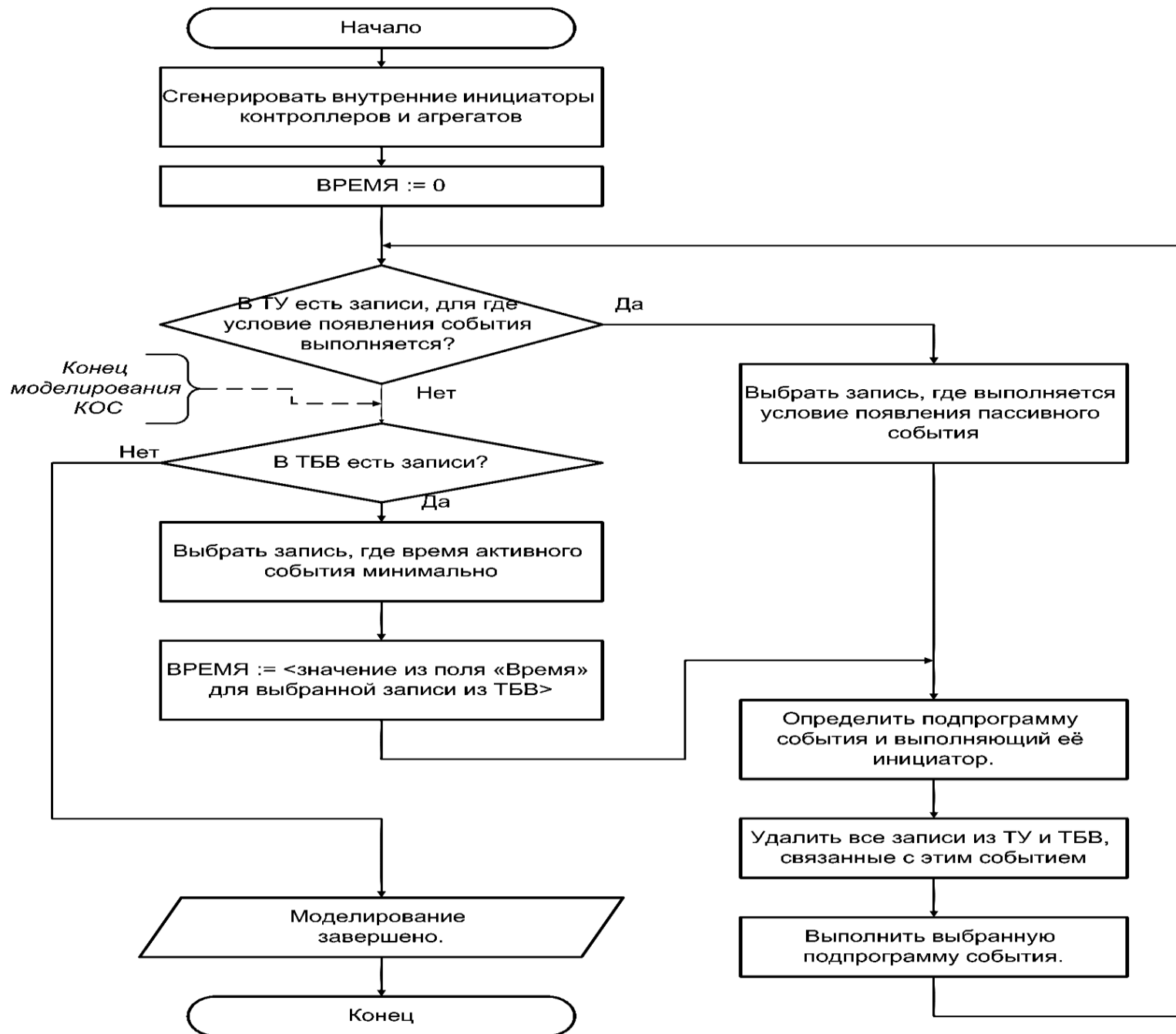
Пассивное событие



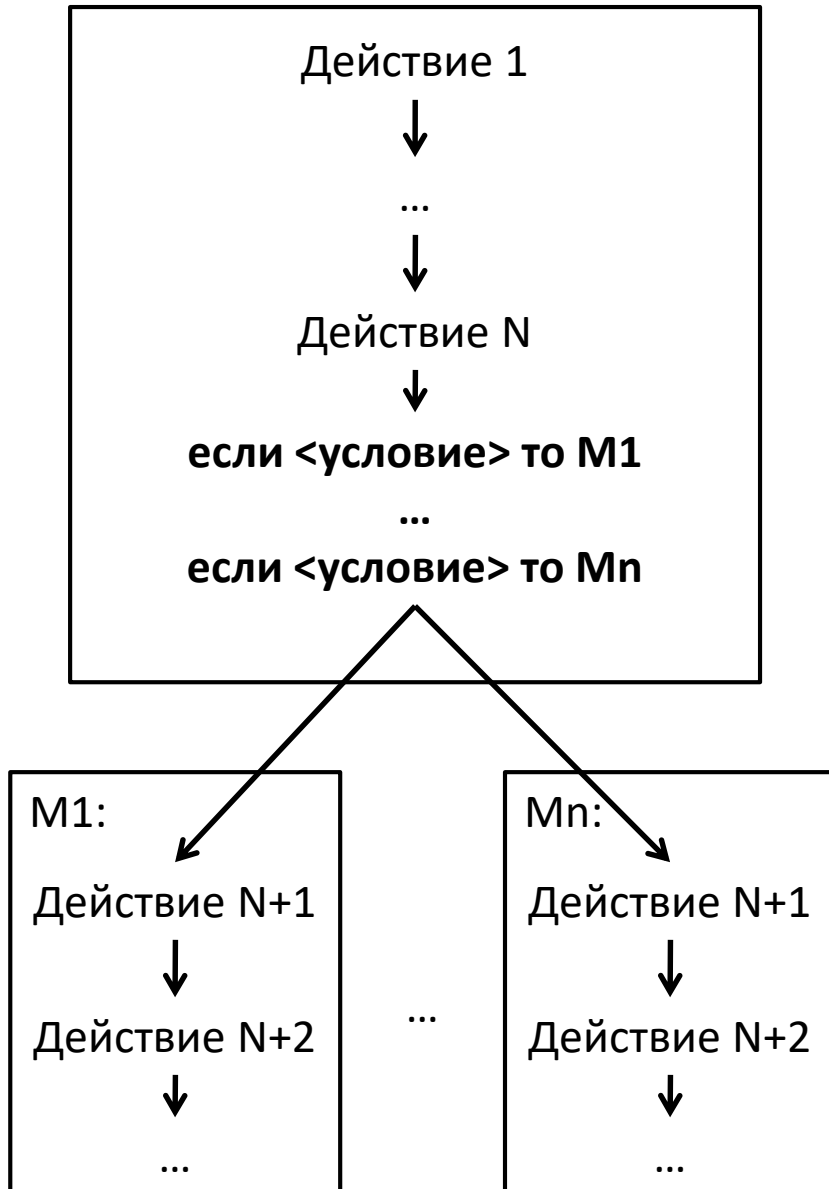
*Подпрограмма
события 1*

*Подпрограмма
события 2*

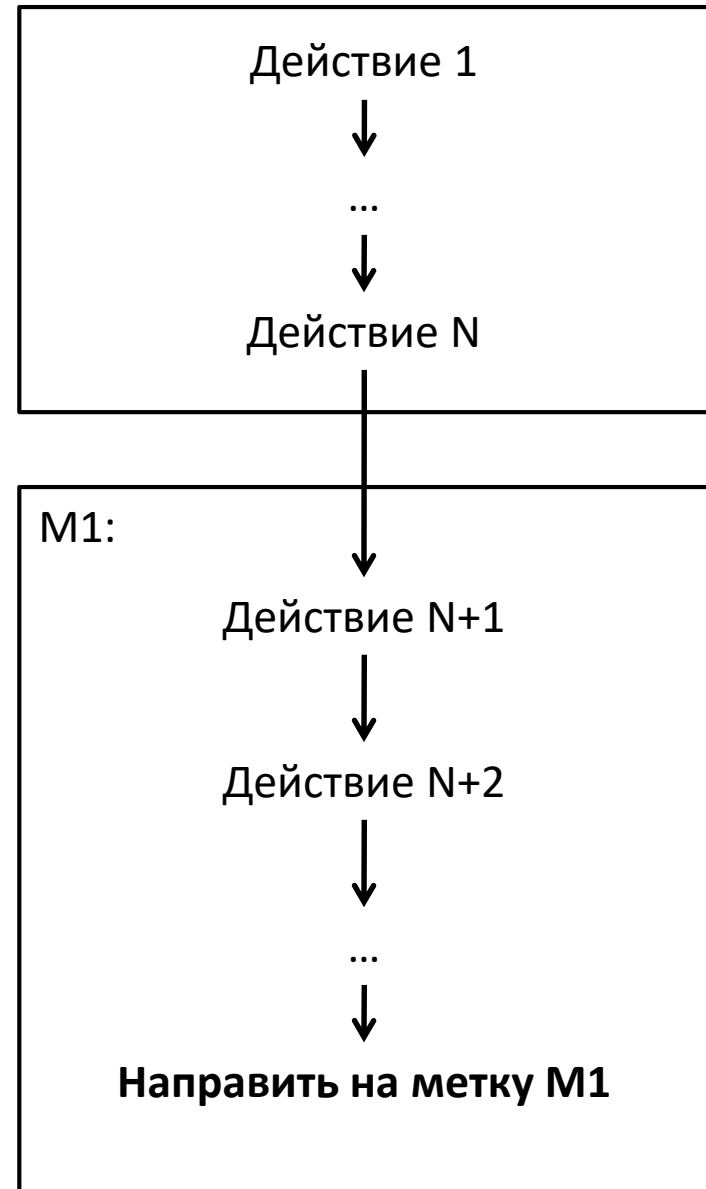
Календарь и АПУ



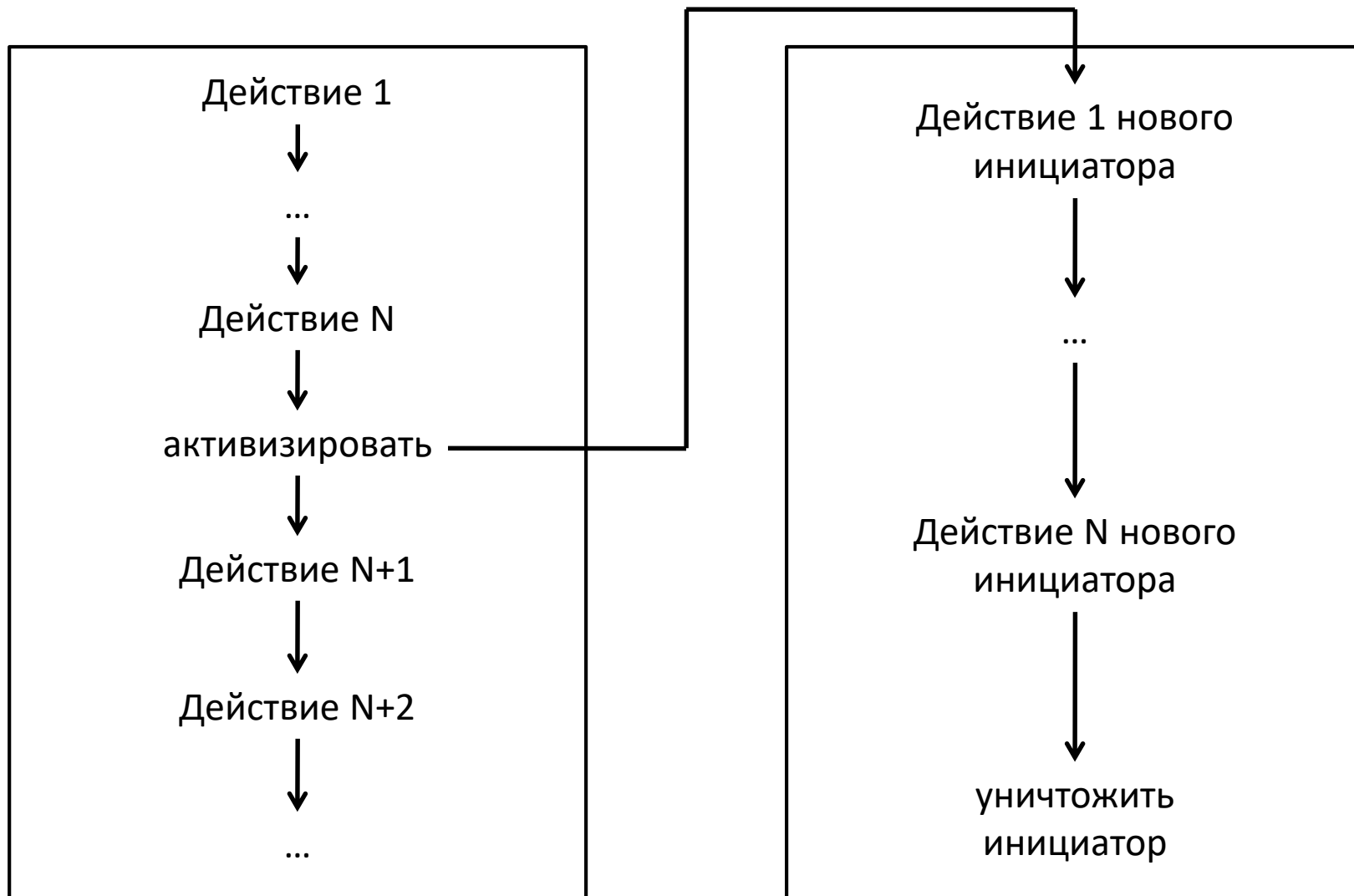
Деление навигационным оператором



Деление меткой



Активизация и уничтожение инициатора



ОРГАНИЗАЦИЯ МОДЕЛИ

Моделирующий алгоритм

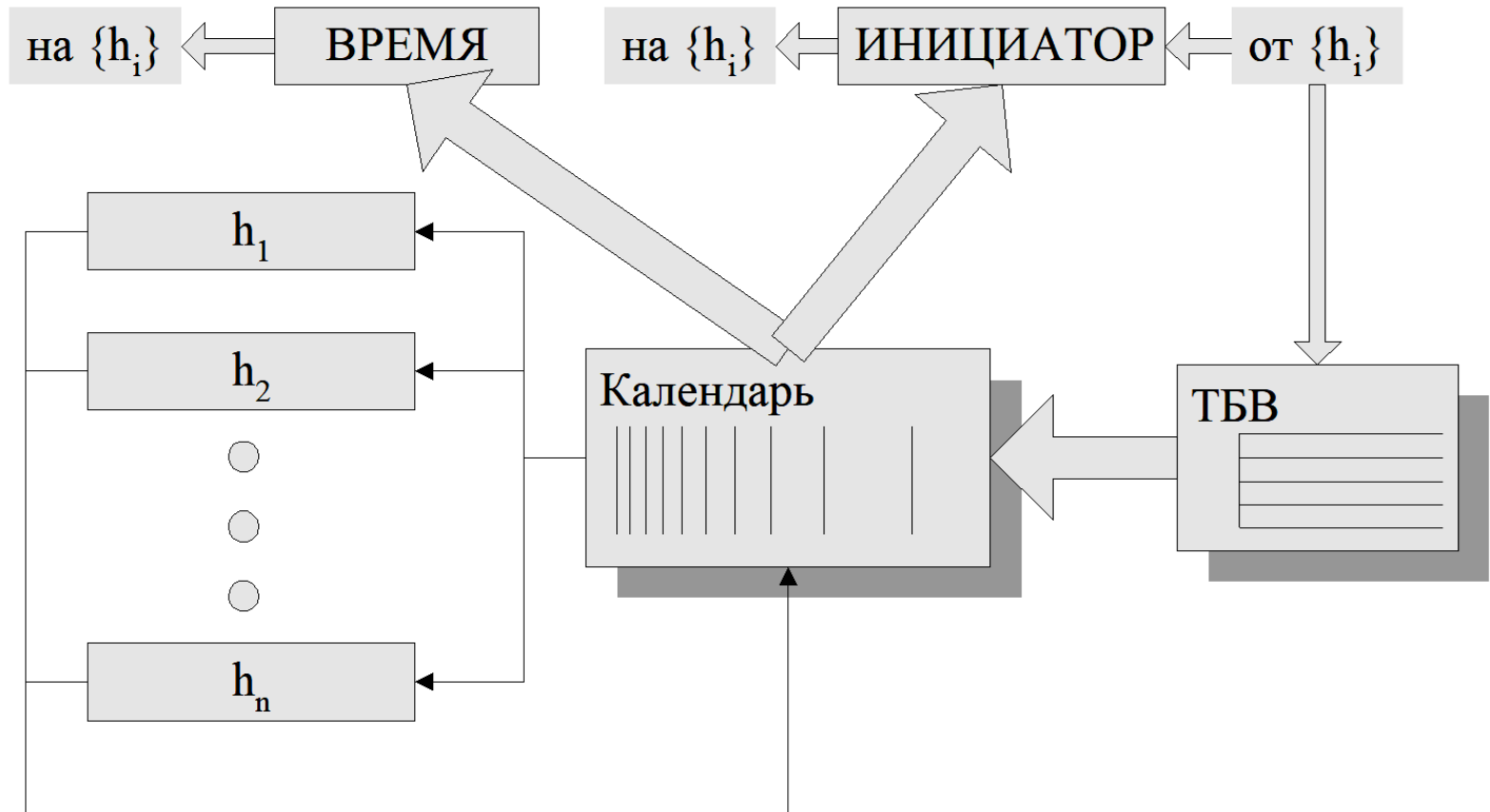
Вычислительная затратность сканирующего алгоритма вызвана необходимостью многократного просчета логических условий в ТУ для автоматизации генерирования КОС.

Чтобы сократить лишние затраты машинного времени на сканирование ТУ, необходимо изменить способ генерирования КОС. Можно процедуры генерации КОС разместить непосредственно в подпрограммах событий и исключить АПУ из моделирующего алгоритма.

Такой модернизированный моделирующий алгоритм назовём

моделирующим алгоритмом линейного типа.

Моделирующий алгоритм линейного типа



ОРГАНИЗАЦИЯ МОДЕЛИ

Моделирующий линейный алгоритм

В линейном алгоритме КАЛЕНДАРЬ выбирает ближайшее активное событие из ТБВ так же, как и в моделирующем алгоритме сканирующего типа.

Затем передаёт управление соответствующей подпрограмме активного события.

Но подпрограмма активного события после своего выполнения передаёт управление той подпрограмме пассивного события, которое должно выполняться в соответствии с графом КОС.

Эта подпрограмма, в свою очередь, передаёт управление следующей по графу КОС подпрограмме пассивного события и т.д., до тех пор, пока не будут исчерпаны все события текущего КОС.

Последняя подпрограмма в текущем КОС передает управление КАЛЕНДАРЮ, что соответствует переходу к новому КОС.

ОРГАНИЗАЦИЯ МОДЕЛИ

МОДЕЛИРУЮЩИЕ АЛГОРИТМЫ

Коэффициент затратности линейного алгоритма близок к 1.

Однако это достигается путем усложнения подпрограмм событий в силу необходимости задавать в них в явном виде все возможные варианты генерации КОС.

Состояние — функция времени, модельное время продвигается событиями, происходящими в системе между двумя соседними моментами времени, соединяющими состояния.

Принципы разбиения системы на состояния, состояний на события и способы перехода системы из одного состояния в другое состояние могут быть самыми различными и зависят от предпочтений и квалификации исследователя, знаний о системе, прикладной задачи.

Функционирование модельной системы выглядит в общем виде так ➔

- 1) система начинает свою работу, модельные часы запущены. Система находится в начальном состоянии S_0 при условном значении модельного времени $T = t_0$;
- 2) последовательно, в соответствии с логикой работы и существующими в системе приоритетами, исполняются действия, соответствующие всем событиям, которые должны произойти при реализации состояния S_0 в момент времени t_0 ;
- 3) определяется следующий ближайший момент времени t_1 , когда должно реализоваться очередное состояние S_1 ;
- 4) формируется список событий, которые должны произойти в ближайший к текущему значению момент времени t_1 и последующим моментам, которые можно спрогнозировать. Это список будущих событий;
- 5) часы модели переводятся на значение t_1 . Оно вычисляется прибавлением к значению времени t_0 значения dt , которое может быть как фиксированным, так и переменным значением;

6) осуществляется перевод всех событий из списка будущих событий, которые должны быть исполнены в момент времени t_1 , в список КОС;

7) производится реализация всех событий из списка КОС. При реализации этих событий могут произойти изменения условий для событий, находящихся в списке КОС и ждущих изменения этого условия. В этом случае осуществляется реализация этих событий, итерация по списку продолжается, пока не будут исполнены все возможные на данный момент t_1 события;

8) формируется новый список *будущих* событий, соответствующий значению момента времени t_2 ;

9) часы модели переводятся на значение t_2 ;

10) повторяется обработка с п.4.