

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РФ
Восточно-Сибирский государственный технологический
университет

Алтаев А.А.

Имитационное моделирование на языке GPSS

МЕТОДИЧЕСКОЕ ПОСОБИЕ ПО ДИСЦИПЛИНЕ
«КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ»

для студентов специальности 220400
«Программное обеспечение ВТ и АС»

Издательство ВСГТУ
Улан-Удэ, 2002

УДК: 681.142

Имитационное моделирование на языке GPSS / Сост. Алтаев А.А. – Улан-Удэ, Изд-во ВСГТУ, 2001. – 122 с.

Методическое пособие включает справочный материал по функциональным возможностям системы имитационного моделирования GPSS с примерами их использования.

Методическое пособие предназначено для студентов специальности 220400 "Программное обеспечение вычислительной техники и автоматизированных систем", изучающих компьютерное моделирование

Рецензент: С.Б.Дарибазарон, к.ф.-м.н., доцент кафедры прикладной математики ВСГТУ

Печатается по решению редакционно-издательского совета ВСГТУ

© ВСГТУ, 2001 г.

ВВЕДЕНИЕ.....	4
1. ОСНОВНЫЕ КОНЦЕПЦИИ СОЗДАНИЯ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ (СМО).....	5
1.1. СИСТЕМА ОБСЛУЖИВАНИЯ С ОДНИМ ПРИБОРОМ И ОЧЕРЕДЬЮ	6
1.2. ОСНОВНЫЕ ПОНЯТИЯ СМО	6
1.3. ЭЛЕМЕНТЫ ПРОЦЕДУРЫ РЕШЕНИЯ.....	7
1.3.1. События	7
1.3.2. Таймер модельного времени	7
1.3.3. Завершение моделирования	8
1.4. АЛГОРИТМИЗАЦИЯ МОДЕЛИ.....	8
1.5. МОДЕЛИРОВАНИЕ МНОГОКАНАЛЬНЫХ УСТРОЙСТВ	9
2. ПРИНЦИП РАБОТЫ СИСТЕМЫ МОДЕЛИРОВАНИЯ ДИСКРЕТНЫХ СИСТЕМ GPSS	9
2.1 ОПИСАНИЕ МОДЕЛЕЙ С ПОМОЩЬЮ ЯЗЫКА GPSS	16
2.2 ОПИСАНИЕ ОБЪЕКТОВ GPSS	21
2.3 БЛОКИ, СВЯЗАННЫЕ С ТРАНЗАКТАМИ GPSS.....	29
2.4 БЛОКИ, ОПИСЫВАЮЩИЕ РАБОТУ ОБОРУДОВАНИЯ	36
2.5 ИЗМЕНЕНИЕ МАРШРУТОВ ТРАНЗАКТОВ	40
2.6 БЛОКИ ДЛЯ СБОРА СТАТИСТИКИ.....	44
2.7 ОСНОВНЫЕ ВЫВОДЫ ПО ЯЗЫКУ GPSS.....	46
ЛИТЕРАТУРА.....	48
ПРИЛОЖЕНИЯ.....	49
ПРИЛОЖЕНИЕ 1. Условные обозначения на блок-диаграммах GPSS	49
ПРИЛОЖЕНИЕ 2. Основные сокращения и обозначения симулятора GPSS.....	54
ПРИЛОЖЕНИЕ 3. Стандартные числовые атрибуты (СЧА) GPSS	55
ПРИЛОЖЕНИЕ 4. Блоки GPSS	58
ПРИЛОЖЕНИЕ 5. Карты GPSS.....	62

ВВЕДЕНИЕ

Моделирование - это метод исследования сложных систем, основанный на том, что рассматриваемая система заменяется на модель и проводится исследование модели с целью получения информации об изучаемой системе. Под моделью исследуемой системы понимается некоторая другая система, которая ведет себя с точки зрения целей исследования аналогично поведению системы. Обычно модель проще и доступнее для исследования, чем система, что позволяет упростить ее изучение. Среди различных видов моделирования, применяемых для изучения сложных систем, большая роль отводится имитационному моделированию. Имитационной называется модель, которая воспроизводит все элементарные явления, составляющие функционирование исследуемой системы во времени с сохранением их логической структуры и последовательности. В последнее время большое прикладное значение получила разновидность имитационного моделирования, в котором в качестве модели используется программа, выполняемая на ЭВМ. Эта разновидность имитационного моделирования называется программным моделированием систем.

Под системой S понимается выделенное в соответствии с некоторым правилом объединение элементов любого рода, образующих связанное целое. Система не изолирована от окружающего мира. Все, что оказывает воздействие на систему, и на что система оказывает воздействие, называется внешней средой. В общем случае состав элементов системы переменный. Одни элементы находятся в системе постоянно, другие - появляются и покидают систему (временные элементы). Все атрибуты элементов и системы в целом можно разделить на два типа: переменные и постоянные. Переменными являются атрибуты, значение которых остается неизменным в рассматриваемом периоде времени.

Совокупность конкретных значений всех переменных атрибутов элементов и системы в целом в некоторый момент времени существования системы определяет состояние системы $z(t)$. Системы в соответствии с различными признаками могут быть классифицированы следующим образом:

- динамические-статические;
- дискретные-непрерывные-комбинированные;
- стохастические (вероятностные)-детерминированные.

Система является динамической, если ее состояние меняется с изменением времени, в противном случае система является статической. Если состояние системы, т. е. значение ее атрибутов, изменяется непрерывно, то она называется непрерывной системой, а если значения изменяются в дискретные моменты времени, то система называется дискретной. Существуют такие системы, у которых часть атрибутов, описывающих состояние системы, меняется непрерывно, а часть дискретно. Эти системы называются непрерывно-дискретными или комбинированными.

Система называется стохастической, если при одних и тех же начальных условиях результаты функционирования системы будут различаться, иначе система называется детерминированной.

Функционирование динамической дискретной системы в период времени $[t_0, T]$ заключается в последовательной смене состояний системы $z(t_1) \rightarrow z(t_2) \rightarrow \dots \rightarrow z(t_n)$, где $t_0 \leq t_1 \leq t_2 \leq \dots \leq t_n \leq T$.

Функционирование системы может рассматриваться и описываться как взаимодействие событий, действий или процессов, происходящих в системе.

Под *событием* понимается всякое изменение состояния системы под воздействием внешней среды и сложившихся в системе условий. Событие рассматривается как мгновенное изменение состояния системы.

Под *действием* понимается пребывание элемента системы в некотором состоянии.

Переход элемента в данное состояние (начало действия) и выход из этого состояния (окончание действия) определяется условиями, сложившимися в системе.

Упорядоченная во времени логически взаимосвязанная последовательность событий, выделенная в соответствии с некоторым признаком, называется процессом. Таким образом, процесс- это более агрегативное понятие, чем событие и действие.

Существует множество систем, процессы функционирования в которых могут быть представлены моделями информационных потоков, получившими название систем массового обслуживания (СМО). Это прежде всего процессы в технических системах – телефонные сети, радиосвязь и телекоммуникации, вычислительные машины, системы и вычислительные сети. При их анализе наиболее важно определить скорость передачи или обработки информации, оценить пропускную способность, загрузку оборудования и т. д. При анализе транспортных систем важнейшими задачами являются определение скорости и объема перевозок, сокращение простоев и др. Процессы жизнедеятельности в биологических системах требуют прежде всего определения благоприятных условий жизни, размножения и развития отдельных особей или популяции (колонии, сообщества) в целом. Многие процессы деятельности человека (социальные, экономические, экологические) могут быть представлены моделями типа СМО. И даже обучение, представляемое как усваивание знаний и забывание, также может быть описано такими моделями.

Любая подобная система неизбежно испытывает различного рода возмущения, источниками которых могут быть либо внешние воздействия, обусловленные случайными или систематическими изменениями окружающих условий, либо внутренние флуктуации, возникающие в самой системе в результате взаимодействия элементов. При исследовании эти системы представляются в виде стохастических моделей дискретных процессов (СМДП). Несмотря на успешное развитие и применение методов аналитического моделирования СМДП, основным методом исследования таких систем остается имитационное моделирование на ЭВМ с применением специализированных языков программирования.

За всю историю развития вычислительной техники было создано более 300 языков моделирования дискретных процессов. Одним из первых языков описания СМДП, появившихся в начале 60-х годов, был язык блок-диаграмм, предложенный Гордоном, идеи которого оказались настолько плодотворны, что использовались во многих последующих разработках в нашей стране и за рубежом. На основе языка блок-диаграмм в 70-х годах был создан и в последующем адаптирован к ПК широко используемый в настоящее время для моделирования большого класса систем язык и система моделирования GPSS (General Purpose Simulation System - Система моделирования общего назначения).

1. ОСНОВНЫЕ КОНЦЕПЦИИ СОЗДАНИЯ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ (СМО)

Система массового обслуживания - это система, состоящая из обслуживающего прибора, заявки, находящейся на обслуживании, и ожидающих обслуживания заявок.

Рассмотрим на процедурном уровне модель системы обслуживания с одним прибором и очередью. Сначала обсудим способ функционирования этой системы и поставим цель разработать модель на ЭВМ, которая промоделировала бы данную систему. Далее рассмотрим основные вопросы, связанные с созданием такой модели. При моделировании таких систем предполагается, что существует генератор случайных чисел. Считают, что обращение к генератору происходит как к функции, которая выдает значения случайных чисел, равномерно распределенных в интервале от 0, 000000 до 0, 999999 включительно.

1.1. СИСТЕМА ОБСЛУЖИВАНИЯ С ОДНИМ ПРИБОРОМ И ОЧЕРЕДЬЮ

Рассмотрим систему, состоящую из одного человека, выполняющего обслуживание определенного вида. Этот человек может быть кассиром, продающим билеты на станции, контролером в универсальном магазине, парикмахером в парикмахерской с единственным креслом. "Клиенты" приходят к такому "обслуживающему прибору" в случайные моменты времени, ждут своей очереди на обслуживание (если есть необходимость), их обслуживают по принципу "первый пришел - первым обслужен". После этого они уходят. Схематично эта ситуация показана на рис. 1.

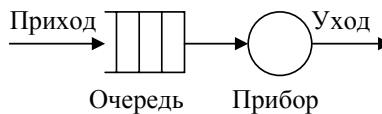


Рис. 1

Здесь необходимо дать пояснения: на рисунке прямоугольник - это обслуживающий прибор, а кружок внутри него - заявка, находящаяся на обслуживании.

1.2. ОСНОВНЫЕ ПОНЯТИЯ СМО

Для дальнейшего рассмотрения системы введем следующие определения:

ОЧЕРЕДЬ - это группа заявок, ожидающих обслуживания.

МОДЕЛЬНОЕ ВРЕМЯ - это промежуток времени между началом моделирования и его завершением.

Простая система массового обслуживания, изображенная на рис. 1, характеризуется двумя независимыми случайными переменными:

ИНТЕРВАЛ ПРИБЫТИЯ заявок - это интервал времени между последовательными моментами прибытия заявок в систему.

ВРЕМЯ ОБСЛУЖИВАНИЯ - это время, требуемое прибору для выполнения обслуживания.

Величины, характеризующие работу данной системы обслуживания и зависящие от двух вышеперечисленных независимых случайных переменных, могут стать предметом исследования. Ниже перечислены некоторые из этих случайных величин:

1. Число заявок, прибывших на обслуживание за заданный промежуток времени.
2. Число заявок, которые попали на обслуживание сразу же по прибытии (минуя очередь).
3. Среднее время пребывания заявок в очереди.
4. Средняя длина очереди.
5. Максимальная длина очереди.
6. Нагрузка прибора, являющаяся функцией времени, которое потрачено прибором на обслуживание в течение заданного промежутка времени.

Следует заметить, что разработку логической схемы модели на ЭВМ, которая будет имитировать систему обслуживания с одним прибором и очередью, нужно вести при следующих условиях:

1. Случайные переменные **ИНТЕРВАЛ ПРИБЫТИЯ** и **ВРЕМЯ ОБСЛУЖИВАНИЯ** являются равномерно распределенными и принимают только целые значения.

2. Все прибывающие заявки должны быть обслужены независимо от длины очереди.

3. Вначале моделирования система "пуста", т. е. нет очереди и обслуживающий прибор свободен.

4. Моделирование продолжается до тех пор, пока не будет достигнуто значение модельного времени, заданное для этой модели в качестве одного из входных данных.

1.3. ЭЛЕМЕНТЫ ПРОЦЕДУРЫ РЕШЕНИЯ

1.3.1. События

При моделировании систем массового обслуживания совершаются некоторые события. Все события в системе должны быть каким-либо образом зафиксированы и должно быть учтено их воздействие на текущее состояние системы. Кроме того, необходимо определить, как нужно корректировать состояние системы в связи с воздействием на нее этих событий. События разделяются на две категории:

1. Основное событие - это такое событие, время возникновения которого можно запланировать заранее. Это, например приход заявки, начало обслуживания и завершение обслуживания.

2. Вспомогательное событие - это событие, время возникновения которого невозможно запланировать заранее. Эти события возникают тогда же, когда и основные, но являются зависимыми, возникающими как следствие основных.

1.3.2. Таймер модельного времени

Так как работа модели связана с последовательным возникновением событий, то вполне естественно использовать понятие "Таймер Модельного Времени" в качестве одного из элементов модели системы. Для этого вводят специальную переменную и используют ее для фиксации текущего времени работы модели.

Опишем теперь некоторые специфические свойства таймера модельного времени.

Когда начинается моделирование, таймер модельного времени обычно устанавливают на нулевое значение. Разработчик сам решает вопрос о том, какое значение реального времени принять за точку отсчета. Например, началу отсчета может соответствовать 8 ч. утра первого моделируемого дня. Разработчик также должен решить вопрос о выборе величины единицы времени. Единицей времени может быть 1 с, 5 с, 1 мин, 20 мин или 1 ч. Когда единица времени выбрана, все значения времени, получаемые при моделировании или входящие в модель, должны быть выражены через эту единицу.

На практике значения модельного времени должны быть достаточно малыми по сравнению с реальными промежутками времени, протекающими в моделируемой системе. В данной системе обычно выбирают единицу времени, равную 1 мин.

Если при моделировании некоторой системы при текущем значении модельного времени ее состояние изменилось, то нужно увеличить значение таймера. Чтобы определить, на какую величину должно быть увеличено значение таймера, используют один из двух методов:

1. Концепция фиксированного приращения значений таймера.

При таком подходе увеличивают значение таймера ровно на одну единицу времени. Затем нужно проверить состояния системы и определить те из запланированных событий, которые должны произойти при новом значении таймера. Если таковые имеются, то необходимо выполнить операции, реализующие соответствующие события, снова изменить значение таймера на одну единицу времени и т. д. Если проверка покажет, что для нового значения таймера не запланировано ни одного события, то произойдет передвижение таймера непосредственно к следующему значению.

2. Концепция переменного приращения значений таймера.

В этом случае условием, вызывающем приращение таймера, является наступление времени "близкого события". Близкое событие - это то событие, возникновение которого запланировано на момент времени, равный следующему ближайшему значению таймера модельного времени. Колебания приращения таймера от случая к случаю объясняют выражение "переменное приращение времени".

Следует заметить, что выгоднее использовать концепцию переменного приращения значений таймера, т. к. при нем можно избежать обработки в промежуточные моменты времени, когда не планируется выполнение никаких событий. В

рассматриваемой задаче с одним обслуживающим прибором и очередью будет использована именно эта концепция. И один из способов определения времени ближайшего события заключается в составлении списка моментов времени, в которые запланированы выполнения различных основных событий. Когда приходит время наращивания значения таймера, в этом списке должно быть найдено ближайшее событие. Если список моментов времени отсортирован в порядке возрастания, то никакого поиска не требуется. Таймер просто установится в нужное значение в соответствии с первой позицией списка. Затем таймеру должно быть придано его значение и управление должно быть передано в ту часть модели, где обслуживается событие, о котором идет речь.

1.3.3. Завершение моделирования

Обычно после какого-то момента времени наступает необходимость прекратить моделирование. Например, нужно предотвратить приход новых заявок в систему, но обслуживание надо продолжать до освобождения системы. Одним из способов является введение в модель основного псевдособытия, называемого "завершением моделирования". Тогда одной из функций модели будет планирование этого события. Момент времени, наступление которого должно вызвать остановку моделирования, задается обычно в виде

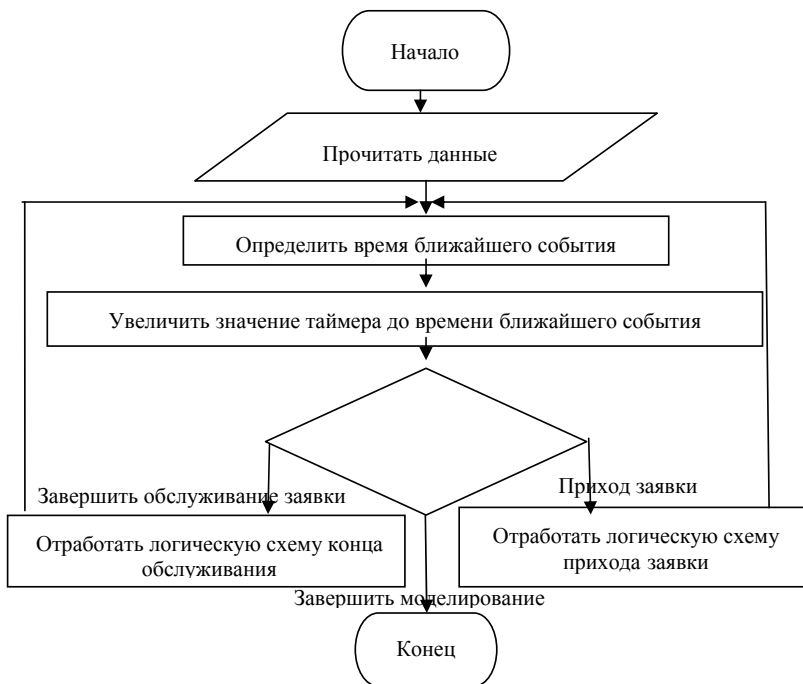


Рис. 2.

числа. Т.о., в процессе моделирования нужно проверять, является ли событие "завершение моделирования" следующим событием. Если "да", то в таймере устанавливается значение времени конца моделирования, а управление передается процедуре, которая обрабатывает завершение моделирования.

1.4. АЛГОРИТМИЗАЦИЯ МОДЕЛИ

Мы рассмотрели различные элементы, из которых формируется модель системы обслуживания с одним прибором и очередью. Последним шагом является сбор этих элементов в единую общую модель. В общем виде логическую схему модели можно представить в виде блок-схемы на рис. 2.

Теперь наша задача состоит в создании машинной модели на ЭВМ, которая позволит изучить поведение системы в течение времени моделирования. Иначе говоря, нужно реализовать эту блок-схему на ЭВМ, используя блоки и операторы языка GPSS.

1.5. МОДЕЛИРОВАНИЕ МНОГОКАНАЛЬНЫХ УСТРОЙСТВ

Прибор в GPSS используют для моделирования единственного устройства обслуживания. Два или более находящихся рядом обслуживающих устройства могут быть промоделированы на GPSS двумя или более приборами, располагаемыми рядом, т. е. параллельно. Этими приборами могут быть, например, два парикмахера, работающих рядом.

GPSS предоставляет для моделирования однородных (обладающих определенными общими свойствами) параллельных приборов специальное средство (или элемент). Для этого используют название "многоканальное устройство" (МКУ). Схематично такая ситуация представлена на рис. 3. В модели может быть несколько многоканальных устройств. Для того чтобы между ними было различие, им можно давать имена. Условия использования имен такие же, как и в случае приборов и очередей. Число приборов, которое моделируется каждым из многоканальных устройств, определяется разработчиком. Здесь употребляют термин "емкость многоканального устройства". Емкость многоканальных устройств определяется с помощью оператора STORAGE.

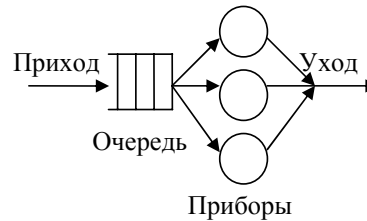


Рис. 3. Три параллельно работающих прибора.

2. ПРИНЦИП РАБОТЫ СИСТЕМЫ МОДЕЛИРОВАНИЯ ДИСКРЕТНЫХ СИСТЕМ GPSS

GPSS является языком моделирования, используемым для построения моделей и проведения моделирования на ЭВМ. Модели на GPSS компактны, часто состоят из меньшего числа операторов, чем такие же модели, написанные на процедурных языках (например, на Паскале или Си). Это объясняется тем, что в GPSS встроено максимально возможное число логических программ, необходимых для моделирующих систем. В него также входят специальные средства для описания динамического поведения систем, меняющихся во времени, причем изменение состояний происходит в дискретные моменты времени. GPSS очень удобен при программировании, поскольку интерпретатор GPSS (здесь и далее интерпретатором называется моделирующая часть системы GPSS) многие функции выполняет автоматически. Например, GPSS без специального на то указания пользователя собирает статистические данные, описывающие поведение модели, автоматически печатает итоговую статистику по завершении моделирования. Пользователю нет необходимости включать в модель операторы для сбора и накопления этих данных или задавать формат, указывающий, в каком виде должны быть распечатаны итоговые данные. В язык включены и многие другие полезные элементы. Например, GPSS обслуживает таймер модельного времени, планирует события, которые должны произойти позднее в течение времени моделирования, вызывает их своевременное появление и управляет очередностью поступления.

Язык моделирования дискретных систем GPSS/PC построен в предположении, что моделью сложной дискретной системы является описание ее элементов и логических правил их взаимодействия в процессе функционирования моделируемой системы. Предполагается также, что для определенного класса моделируемых систем можно выделить небольшой набор абстрактных элементов — объектов. Причем набор

логических правил также ограничен и может быть описан небольшим числом стандартных операций. Комплекс программ, описывающих функционирование объектов и выполняющих логические операции, является основой для создания программной модели системы данного класса.

Описание системы на GPSS представляет собой последовательность блоков, каждый из которых соответствует некоторому оператору (подпрограмме). Каждый блок имеет определенное количество реквизитов, называемых полями, которые отделяются друг от друга запятой (это аналоги параметров процедур и функций в языках программирования), но положение полей строго фиксировано, и отсутствие некоторого поля отмечается запятой. Блоки записываются в следующем формате:

номер_строки метка имя_операции операнды ; комментарии

Строки нумеруются от 1 до 9999999. Операторы располагаются в программе в соответствии с нумерацией строк. Некоторые операторы не требуется включать в исходную программу. Такие операторы вводятся без номера строки. Отдельные операторы могут иметь метку для ссылки на эти операторы в других операторах. Между номером строки и меткой ставится один пробел. Отсутствие метки также обозначается пробелом.

Имя операции - это ключевое слово, указывающее конкретную функцию, выполняемую данным оператором. Операнды - уточняющие и конкретизирующие выполнение функции, определённой в поле операции. Эти поля в зависимости от типа операции содержат до девяти операндов, расположенных в определённой последовательности и обозначаемыми буквами латинского алфавита от A до I. Некоторые операции вообще не имеют операндов, а в некоторых операнды могут быть пропущены. Если некоторое частичное поле остается пустым, т. е. не нужен соответствующий аргумент, это отмечается еще одной запятой, например: PRINT,, MOV¹. Здесь используется только аргумент MOV в частичном поле C (A и B пустые).

В качестве значений, записываемых в указанные поля (A...I), можно использовать: а) некоторый стандартный числовой атрибут (СЧА); б) положительную константу Kn ; в) специальные обозначения, определяющие режим (например, ALL для блока TRANSFER); г) определенные СЧА (только Xn , Vn и т. д.).

В GPSS имеется ряд средств для внесения комментариев в тело программы. Во-первых, наличие символа (*) в первой колонке указывает на то, что карта содержит только комментарии и не будет просматриваться транслятором. Во-вторых, появление символа (;) в списке операндов означает окончание списка и начало комментария. Информация, следующая после (;) в поле операндов, может содержать любые символы русского и латинского алфавитов и не подлежит просмотру транслятором.

Каждый оператор относится к одному из четырёх типов:

- карты описания блоков (формируют логику модели),
- карты описания объектов (описание параметров объектов),
- управляющие карты (управление процессом моделирования),
- управляющие карты системы (управление интегрированной средой GPSS/PC).

После трансляции исходной программы в памяти ПК создаётся текущая модель, являющаяся совокупностью разного типа объектов. Каждому объекту соответствуют числовые атрибуты, описывающие его состояние в данный момент модельного времени. Значения атрибутов могут быть арифметическими или логическими. Атрибуты, доступные программисту, называются *стандартными числовыми атрибутами* (СЧА) (см. Приложение 3).

Кроме того, имеется ряд так называемых системных атрибутов, относящихся не к отдельным объектам, а к модели в целом. Значение атрибутов всех объектов модели по окончании моделирования выводятся в стандартный отчёт. Все СЧА являются целыми

¹ пробелы недопустимы

числами². Для ссылки на стандартные числовые атрибуты соответствующий операнд оператора записывается, так:

СЧА\$имя

или так:

СЧА_{*i*}

где СЧА - конкретный СЧА, имя - имя объекта, *i* - номер объекта, \$ - символ-разделитель. Каждый объект GPSS имеет имя и номер. Имя объекта представляет собой начинающуюся с буквы последовательность букв латинского алфавита, цифр и символа подчёркивания "_", общее число символов – от трех до пяти. Первые три символа – буквы. При необходимости имени любого объекта, кроме имени блока, можно поставить в соответствие любой номер с помощью карты EQU, имеющей следующий формат:

Метка	Операция	A
j	EQU	номер

где j – имя объекта

Для того чтобы показать пошаговое выполнение процедур, было использовано символическое изображение передач управления в виде специальных фигур и линий (см. рис. 2.). Управление передавалось от одного блока к другому (или от одного действия к другому). Однако в GPSS концепция "передачи управления от блока к блоку" имеет специфические особенности и требует более подробного рассмотрения. Конфигурация блок-схемы GPSS-модели отражает направления, по которым происходит движение перемещающихся элементов. Каждый такой элемент называется **транзактом**. Транзакты являются динамическими (т. е. движущимися) элементами GPSS-модели. Работа этой модели заключается в перемещении транзактов от блоков к блокам. Некоторые примеры возможных аналогий между транзактами и элементами реальных систем представлены в табл. 1.

Таблица.1

Системы	Элементы систем, символизируемые транзактами
Магазин	Покупатель
Бензозаправка	Автомобиль
Дисплейный класс	Компьютеры
Парикмахерская	Клиент

Таким образом, перемещение транзакта от блока к блоку в модели аналогично, например, передвижению клиента в парикмахерской от одной стадии к другой.

В самом начале моделирования в GPSS-модели нет ни одного транзакта. В процессе моделирования транзакты входят в модель в определенные моменты времени в соответствии с теми логическими потребностями, которые возникают в моделируемой системе. Подобным же образом транзакты покидают модель. В общем случае в модели существует большое число транзактов, но в один момент времени движется только один.

Если транзакт начал свое движение, он перемещается от блока к блоку по пути, предписанному блок-схемой. Такое продвижение транзакта продолжается до тех пор, пока не произойдет одно из следующих возможных событий:

- транзакт входит в блок, функцией которого является удаление транзакта из модели,
- транзакт входит в блок, функцией которого является задержка транзакта на некоторое время,
- транзакт "пытается" войти в следующий блок в соответствии с блок-схемой, но блок

² Атрибуты, имеющие дробную часть, имеют значения, выраженные в долях тысячи. Например, если прибор находился в работе 63,4% времени моделирования, то его коэффициент нагрузки равен 0,634, а его атрибут Fgi = 634

"отказывается" принять его.

Если возникло одно из описанных условий, то транзакт остается на месте, и начинается перемещение в модели другого транзакта. Таким образом, выполнение моделирования в системе продолжается.

Центральной задачей, выполняемой симулятором, является определение того, какой транзакт надо выбрать следующим для продвижения в модели, когда его предшественник прекратил продвижение. С этой целью симулятор рассматривает каждый транзакт как элемент некоторого списка. В относительно простых моделях используются список текущих событий (время продвижения меньше или равно модельному) и список будущих событий (время продвижения больше модельного). Симулятор имеет таймер модельного времени. Таймер имеет следующие особенности:

- регистрируются только целые значения,
- единица модельного времени выбирается разработчиком,
- симулятор не анализирует состояние модели в каждый следующий момент модельного времени, а продвигает таймер к моменту времени, когда происходит ближайшее следующее событие.

Симулятор GPSS/PC помещает транзакты в зависимости от условий в модели в тот или иной список и переносит их из списка в список, просматривает списки, выбирает следующий транзакт для обработки, корректирует таймер модельного времени после обработки всех транзактов в списке текущих событий.

Модель на GPSS состоит из одного или нескольких независимых сегментов. В процессе моделирования активным является тот из сегментов, в котором находится перемещающийся в настоящий момент транзакт. Когда он блокируется, начинает двигаться следующий транзакт, и может быть так, что этот следующий транзакт принадлежит другому сегменту модели. Таким образом, происходит переключение активности между сегментами.

Объекты GPSS подразделяются на 7 категорий и 14 типов (табл. 1) и позволяют, с одной стороны, описать их взаимодействие сравнительно несложными наборами операций, с другой — достаточно просто и наглядно представить процесс функционирования исследуемой системы S, формализуемой в виде Q-схемы.

Таблица 2

Категория	Типы объектов
Динамическая	Транзакты
Операционная	Блоки
Аппаратная	Устройства Памяти Ключи
Вычислительная	Переменные: <ul style="list-style-type: none">• арифметические,• булевские Функции
Статистическая	Очереди Таблицы
Запоминающая	Матрицы ячеек Ячейки
Группирующая	Списки пользователя Группы

Для облегчения пользователю процесса построения модели в GPSS разработан так называемый язык *блок-диаграмм* (см. приложение 1, где блоки приведены в порядке латинского алфавита), позволяющий упростить переход от алгоритма к программе модели системы S. Каждый блок GPSS имеет свой графический аналог, с помощью которых отображается пространственная конструкция модели, упрощая дальнейшую линейризацию программы модели.

Рис. 4. Процесс создания модели с использованием GPSS

Построение блок-диаграмм знакомит программиста с набором операторов языка. Набор операторов языка однозначно соответствует набору блоков для описания блок-диаграммы. Вследствие этого очевидно, что построение блок-диаграммы является не самоцелью, а лишь промежуточным этапом при написании имитационной модели исследуемой системы с использованием операторов языка GPSS/PC. При этом процесс создания модели можно изобразить в виде схемы, показанной на рис. 4.

Основные сокращения и обозначения, используемые симулятором GPSS на этапе системного ввода, приведены в приложении 2.

Динамическими объектами являются транзакты (сообщения), которые представляют собой единицы исследуемых потоков и производят ряд определенных действий,

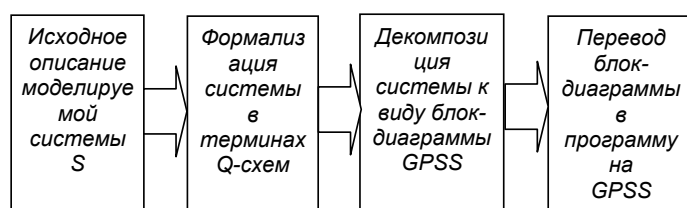


Рис. 4. Схема создания модели

продвигаясь по фиксированной структуре, представляющей собой совокупность объектов других категорий.

Операционные объекты, т.е. блоки, задают логику функционирования модели системы и определяют пути движения транзактов между объектами аппаратной категории.

Объекты аппаратной категории — это абстрактные элементы (устройства памяти и логические ключи), на которые может быть декомпозировано оборудование реальной системы. Воздействуя на эти объекты, транзакты могут изменять их состояние и влиять на движение других транзактов.

Вычислительная категория служит для описания таких ситуаций в процессе моделирования, когда связи между компонентами моделируемой системы S наиболее просто и компактно выражаются в виде математических (аналитических и логических) соотношений. Для этих целей в качестве объектов вычислительной категории введены арифметические и булевские переменные и функции.

К *статистическим объектам* относятся очереди и таблицы, вводимые для оценки характеристик поведения системы.

В процессе моделирования системы одни объекты взаимодействуют с другими, в результате чего происходят изменения атрибутов и преобразование арифметических или логических значений их. Такие преобразования называются *событиями*.

Транзакты моделируют прохождение по системе соответствующих единиц исследуемого потока. Такое движение может быть разбито на цепь элементарных событий, происходящих в определенные моменты времени. Основной задачей симулятора GPSS является определение моментов наступления этих событий,

расположение их в правильной временной последовательности и выполнение соответствующих действий при наступлении каждого события. Чтобы обеспечить правильную последовательность обработки событий во времени, в симуляторе GPSS имеются «системные часы», хранящие значения абсолютного времени модели. Все отрезки времени описываются целыми числами. Поэтому перед составлением модели необходимо провести временное масштабирование для всех временных параметров и характеристик модели системы.

Следующие СЧА не связаны только с одним отдельным объектом, а используются моделью в целом: C1—текущее значение относительного времени с начала процесса моделирования; Kn или n — положительная константа; RNx —одно из восьми случайных чисел ($1 \leq x \leq 8$), значения которых находятся в пределах 0...999.

В GPSS имеется два основных типа объектов: транзакты и блоки, относящиеся соответственно к динамической и операционной категории. Практически все изменения состояний модели системы S происходят в результате входа транзактов в блоки и выполнения блоками своих функций. С блоками непосредственно связаны: операционные блоки, изменяющие процесс моделирования; блоки вывода на печать промежуточных результатов моделирования; карты, управляющие процессом моделирования; карты, управляющие редактированием результатов моделирования. Всем блокам GPSS присваиваются порядковые номера, карты не нумеруются.

Событийное моделирование в GPSS реализуется следующим образом.

- **Организация модельного времени.** Одним из базовых понятий в СМО является событие, определяемое как изменение состояния системы. Моделирование системы на GPSS, по существу, представляет собой имитацию последовательности переходов системы из одного состояния в другое в некоторые моменты времени, поэтому важнейшим является организация моделирования во времени. Время в GPSS-модели отображается в условных единицах (целыми числами), масштаб которых устанавливается исследователем. Приращение модельного времени на каждом шаге моделирования - величина случайная, равная интервалу времени между последовательными событиями.

- **Обработка списков транзактов.** Для имитации процессов, протекающих в моделируемой системе, в GPSS предусмотрены следующие механизмы: - все транзакты, порождаемые в процессе моделирования, образуют списки, в которых транзакты отсортированы, во-первых, по времени, во-вторых, при равных временах у транзактов, по приоритетам; - в процессе моделирования может быть образовано до пяти списков: *список текущих событий* содержит транзакты, которые должны перемещаться в текущий момент модельного времени; все они имеют одинаковое время, равное текущему, и если их приоритеты совпадают, то очередной транзакт для перемещения выбирается по правилу FIFO³; *список будущих событий* содержит транзакты, которые будут перемещаться по модели в будущие моменты времени; в этот список попадают транзакты, вошедшие в блоки имитации задержки или заблокированные по каким-либо условиям (типа светофора или "пробки" для автомобиля); *списки прерываний, синхронизации и пользователя* содержат транзакты, обслуживание которых прервано по некоторым условиям, задаваемым блоками прерывания, синхронизации или находящимся под управлением пользователя.

- **Алгоритм событийного моделирования** состоит в формировании, просмотре и изменении этих списков, в перемещении транзактов из одного списка в другой и в продвижении одного из транзактов (активного, находящегося на вершине списка текущих событий) по блокам модели.

Продвижение текущего транзакта продолжается по блокам модели до тех пор, пока не произойдет одно из следующих событий:

➤ транзакт входит в блок задержки ADVANCE A,B, в котором время транзакта

³ First In – First Out (первым вошел – первым вышел)

увеличивается на значение, определяемое полями А и В так же, как и в блоке GENERATE, и транзакт переходит в список будущих событий;

- транзакт входит в один из блоков проверки условий типа if...then...else (GATE_R А, В или TEST_T А, В), и условие не позволяет транзакту перемещаться дальше (наступает условие блокировки), тогда транзакт переводится в список будущих событий;
- транзакт входит в блок удаления TERMINATE.

Затем выбирается из списка текущих событий следующий транзакт и начинается его продвижение по модели. Если становится невозможным продвижение всех транзактов из списка текущих событий, то изменяется текущий момент времени (т. е. наступает время следующего события или группы событий) и все сказанное выше повторяется.

Транзакты представляют собой описание динамических процессов в реальных системах. Они могут описывать реальные физические объекты, например автомобили в очереди у бензозаправочной колонки.

Кроме того, транзакты могут описывать и нефизические (программные) объекты, например каналные программы, порядок выбора в коммутационных схемах и т. д. Транзакты можно «генерировать» и «уничтожать» в процессе моделирования системы S.

Основным атрибутом любого транзакта являются его параметры, число которых для каждого транзакта может быть от 0 до 1020.

Параметры обозначаются как PX_i^f , где i — номер параметра, X — тип параметра (X — может принимать значения в зависимости от форматов: F — «слово», H — «полуслово», B — «байт», L — «плавающая точка»).

Использование параметров определяется пользователем GPSS. Например, при моделировании сети передачи данных программист может резервировать первый параметр для значения длины сообщения, второй — для номера источника (терминала), выдавшего это сообщение, и т. д.

Важным атрибутом любого транзакта является уровень приоритета PR, изменяющийся в пределах от 0 до 127. В случае, когда два транзакта соперничают при занятии данного устройства, сначала обрабатывается тот, у которого приоритет выше. Если приоритеты у обоих транзактов одинаковые, сначала обрабатывается тот, у которого время ожидания обработки больше. В одном задании может выполняться как один, так и несколько прогонов одной модели. При этом текущим значением абсолютного времени модели AC1 будет называться суммарное время по всем реализованным прогонам, а текущим значением относительного времени модели C1—системное время в пределах одного прогона.

Время, в течение которого транзакт обрабатывается в процессе моделирования, обозначается M1 и называется *транзактным временем*. Оно определяет интервалы времени: 1) с момента относительного времени; 2) с момента прохода транзакта через блок MARK до текущего момента относительного времени.

Параметрическое транзактное время MP_n вычисляется вычитанием из текущего момента относительного времени значения n -го параметра данного транзакта типа PX.

Моделирование и синхронизация параллельных процессов обеспечивается как механизмами и средствами продвижения транзактов по модели, так и дополнительными средствами размножения и синхронизации во времени:

- программа моделирования может состоять из нескольких сегментов, в каждом из которых транзакты порождаются и перемещаются независимо от других сегментов;
- существует блок копирования (размножения) семейства транзактов SPLIT A, B, C, D (где A - количество копий транзакта; B - блок, в который переходят копии транзактов; C - параметр, в котором хранятся номера копий транзактов; D - количество параметров, задаваемых копиям транзактов) с последующим перемещением их по ветвям модели и сборкой либо в блоке GATHER Ag (Ag - количество собираемых

⁴ вторая форма записи - P_i

копий) с последующим перемещением по правилу FIFO, либо в блоке ASSEMBLE Аа с объединением Аа-транзактов в один транзакт для последующего его продвижения;

➤ существуют блоки синхронизации перемещения транзактов по модели MATCH А (здесь А - имя блока, сопряженного с данным), и поступивший в него транзакт продолжит перемещаться только при условии поступления некоего другого транзакта в сопряженный блок.

Изменение последовательного перемещения транзакта по модели может быть нарушено оператором блока TRANSFER, определяющим для данного транзакта номер следующего блока.

Перечислим все блоки GPSS и остановимся более подробно на тех, которые понадобятся для составления программ при выполнении курсовой работы.

Блоки GPSS используются для описания функций моделируемой системы S и управляют движением транзактов. Практически все изменения состояния модели системы S возникают в результате поступления транзактов в соответствующие блоки и выполнения подпрограмм, связанных с этими блоками. После выполнения соответствующей подпрограммы транзакт либо продолжает движение к следующему блоку, либо задерживается на некоторое время в блоке.

У каждого блока имеется два СЧА: Wn — счетчик входов в блок или ожидающий счетчик, который содержит в себе номер текущего транзакта, находящегося в блоке n ; Nn — общий счетчик транзактов, поступивших в блок с начального момента моделирования или с момента обнуления (с помощью карт RESET либо CLEAR). Оба счетчика меняют свое содержимое автоматически.

В зависимости от назначения блоки подразделяются на несколько групп (см. Приложение 4):

1. Блоки, осуществляющие модификацию атрибутов транзактов: а) временная задержка ADVANCE; б) генерирование и уничтожение транзактов GENERATE, TERMINATE, SPLIT, ASSEMBLE; в) синхронизация движения нескольких транзактов MATCH, GATHER; г) изменение параметров транзактов ASSIGH, INDEX, MARK; д) изменение приоритета PRIORITY.

2. Блоки, изменяющие последовательность передвижения транзактов (блоки передачи управления): TRANSFER, LOOP, TEST, GATE.

3. Блоки, связанные с группировочной категорией: JOIN, REMOVE, EXEMINE, SCAN, ALTER.

4. Блоки, организующие использование объектов аппаратной категории: а) устройства (технические средства) SEIZE, RELEASE, FAVAIL, PREEMPT, RETURN, FUNAVAIL, б) памяти (запоминающие устройства) ENTER, LEAVE, SAVAIL, SUNAVAIL, в) ключи (логические переключатели) LOGIC.

5. Блоки, сохраняющие необходимые значения для дальнейшего использования: SAVEVALUE, MSAVEVALUE.

6. Блоки, обеспечивающие получение статистических результатов; а) очереди QUEUE, DEPART; б) статистические таблицы TABULATE, TABLE.

7. Специальные блоки BUFFER, PRINT, EXECUTE, COUNT'X', CHANGE, TRACE, UNTRACE, SELECT'X', HELP.

8. Блоки для организации цепей: LINK, UNLINK.

9. Вспомогательные блоки: WRITE, SAVE, LOAD, REPORT, UPDATE.

2.1 ОПИСАНИЕ МОДЕЛЕЙ С ПОМОЩЬЮ ЯЗЫКА GPSS

В дополнение к различным картам блоков и картам-описаниям для задания дополнительной информации симулятору GPSS требуются определенные управляющие карты (см. Приложение 5).

Карта SIMULATE. Если нужно провести счет по составленной модели системы, то должна быть карта SIMULATE. Допускается две формы заполнения этой карты:

SIMULATE и SIMULATE m , где m — число минут машинного времени, по истечении которого моделирование будет прекращено. Например, оператор SIMULATE 5 определяет длительность счета модели, равную 5 мин. По истечении этого времени моделирование прекращается и на печать выдается накопленная к этому моменту статистическая информация. Если карты SIMULATE нет, то работа (JOB) будет закончена после окончания ассемблирования.

Карта START. Эта карта показывает симулятору GPSS, что все входные данные уже получены и можно начинать счет. Поле А определяет число транзактов, которые должны пройти через систему до выдачи окончательной статистики, называемой счетчиком числа завершений и обозначаемой TG1. Счет заканчивается в тот момент, когда $TG1 < 0$. Значение TG1 уменьшается на величину, заданную в поле А блока TERMINATE, при каждом входе сообщения в этот блок. Если требуется закончить моделирование после прохождения через модель 1000 транзактов, выводимых с помощью карты TERMINATE с единицей в поле А, то в поле А карты START записывается число 1000. Например:

```
START      1000
```

Поле В карты START может быть использовано для блокирования выдачи статистики в конце счета. Для этого нужно записать в поле В мнемоническое обозначение NP. При наличии такой записи в поле В сбор статистической информации в процессе счета не прекращается, блокируется только печать собранной информации по окончании счета. Если поле В пусто, печать происходит как обычно. Эта возможность часто полезна в том случае, когда модель запускается, но пользователь не хочет собирать статистику о начальном периоде счета. В этом случае за первой картой START следует вторая с несколько большим значением счетчика завершения для обычного счета с выдачей информации. Например, если требуется выдавать результаты моделирования через 400 и 600 единиц времени, то можно использовать следующий программный сегмент:

```
GENERATE   200
TERMINATE  1
START      1,NP
START      2,,1
```

Поле С карты START можно использовать для задания интервала промежуточных выдач статистик. Счетчик интервала уменьшается на то же число единиц, что и счетчик завершения. Когда значение этого счетчика становится равным нулю или отрицательным, то производится выдача статистики в обычном формате и восстановление счетчика, после чего счет продолжается. Например, если требуется остановить модель через 600 единиц времени и через каждые 200 единиц выдавать результаты, то можно записать следующий программный сегмент:

```
GENERATE   200
TERMINATE  1
START      3,,1
```

Поле D может содержать указание на то, что при выдаче информации (как в конце выдачи промежуточных результатов, так и в конце счета) нужно распечатывать списки текущих событий, будущих событий, список прерываний, а также списки пользователя. Значение «1» в поле D указывает на то, что пользователю нужна печать списков. В противном случае поле оставляют незаполненным.

Пример 1. Построить программу модели процесса прохождения 500 деталей, поступающих в промежутке времени 11 ± 5 единиц, распределенном по равномерному закону. Детали обрабатываются одним рабочим в течение 10 ± 7 единиц времени» распределенного по равномерному закону. После прохождения 300 деталей требуется выдавать статистику с интервалом в 25 деталей. Программа имеет вид (здесь и в других примерах моделирования справа дается комментарий):

```
SIMULATE      Начать имитацию
```

GENERATE	11,5	Поступление деталей
SEIZE	1	Запуск
ADVANCE	10,7	Обработка
RELEASE	1	Выпуск
TERMINATE	1	
START	275, NP	Прогон без статистики
START	225,,25	Интервальные статистики
END		

Пользователю может понадобиться провести серию просчетов, т. е. несколько просчетов один за другим. Для облегчения задания такого режима введены три типа управляющих карт, в различной степени возвращающих модель к исходному состоянию: карта RESET, стирающая всю накопленную статистику, карта CLEAR, которая стирает всю статистику и все транзакты, и карта JOB, стирающая всю статистику, все транзакты и все блоки.

Карта RESET. Результатом действия этой карты является то, что стирается вся статистическая информация, накопленная в процессе моделирования. Счетчики числа входов в блоки сбрасываются до «0», но содержимое счетчиков ожидания не изменяется. Коэффициенты использования устройств сбрасываются до «0», счетчики числа входов в устройство устанавливаются в соответствии с текущим состоянием устройства. Счетчик равен «0», если устройство свободно, «1» — если оно занято или обслуживание сообщения прервано, и «2» — если устройство занято и обслуживает прерывание. Коэффициенты использования памяти также сбрасываются до «0», а счетчики числа входов в память устанавливаются в соответствии с текущим содержимым каждой памяти. Коэффициенты использования очередей сбрасываются до «0», и счетчики числа входов устанавливаются равными текущей длине соответствующих очередей. Счетчики всех нулевых задержек сбрасываются до «0», а максимальные значения длины очереди устанавливаются равными текущей длине очереди. Все счетчики числа наблюдений интервалов таблиц, все средние значения и т. д. также сбрасываются до «0». Содержимое ячеек SAVEVALUE и состояние логических переключателей не изменяются. Датчик случайных чисел не возвращается к исходному числу.

Пользователь может также задать объекты (устройства, памяти, очереди и др.), атрибуты которых останутся без изменения после выполнения операций, соответствующих карте RESET.

Пример 2. Построить программу модели процесса прохождения деталей, поступающих через интервалы времени, распределенные равномерно в промежутке 7...11 единиц. Детали направляются к одному рабочему, обрабатывающему их со временем 8 ± 3 единицы, распределенным по равномерному закону. Прохождение деталей моделируется по трем интервалам, каждый на 480 единиц времени. После каждого интервала выдается вся статистика, затем вся информация стирается, кроме содержимого устройства и информации о нем. Для этого примера программа имеет вид:

SIMULATE		Начать моделирование
GENERATE	9,2	Генерация деталей
SEIZE	OBR	Запуск на обработку
ADVANCE	8.3	Обработка
RELEASE	OBR	Окончание обработки
TERMINATE		
GENERATE	480,,480	
TERMINATE	1	
START	1	Прогон 1
RESET	F1	
START	1	Прогон 2

RESET	F1	
START	1	Прогон 3

Карта CLEAR. В результате действия этой карты стирается вся накопленная в процессе моделирования статистика и из системы удаляются все транзакты. Счетчики числа входов и счетчики ожидания всех блоков сбрасываются до «0». Коэффициенты использования всех устройств и счетчики числа входов сбрасываются до «0» и все устройства переводятся в состояние «не занято». Коэффициенты использования памяти, счетчики числа входов и текущее содержимое сбрасывается до «0» для всех памяти.

Коэффициент использования очередей, счетчики числа входов, счетчики числа нулевых задержек, текущая и максимальная длина очереди сбрасываются до «0». Стирается вся информация в таблицах и ячейках SAVEVALUE. Все логические переключатели устанавливаются в «0», из системы удаляются все транзакты, а абсолютное и относительное время устанавливается в «0». Затем запускаются все блоки GENERATE. Начальные интервалы задержки, заданные в поле C блоков GENERATE, отсчитываются после завершения операций карты CLEAR. Счетчикам пределов, заданным в поле D, присваиваются их исходные значения. Работа этих счетчиков также начинается после выполнения операций карты CLEAR. Операции этой карты производятся в момент ее появления, причем действие карты CLEAR на следующие за ней карты входной колоды не распространяется. Перед первой картой START модели карта CLEAR не нужна. Датчик случайных чисел не возвращается к исходному числу.

В GPSS операции карты CLEAR расширены для того, чтобы пользователю дать возможность указать отдельные ячейки SAVEVALUE (или ряд таких ячеек), значения которых не будут стерты картой CLEAR.

Пример 3. Построить программу модели процесса прохождения деталей, поступающих со временем, распределенным равномерно в интервале 12... 24 единицы. Детали обрабатываются одним рабочим со временем 16 ± 3 единицы, распределенным по равномерному закону. Прохождение деталей моделируется по четырем интервалам на 480 единиц времени каждый. Подсчитывается количество деталей, изготовленных в каждом интервале времени, и нарастающая сумма изготовленных деталей по всем интервалам. После каждого интервала требуется вывод статистической информации о модели. Моделирование в интервалах независимо друг от друга. Для приведенного примера программа имеет вид:

SIMULATE		Начало моделирования
GENERATE	18,6	Поступление деталей
SEIZE	1	Запуск деталей
ADVANCE	16,3	Обработка
RELEASE	1	Выпуск деталей
SAVEVALUE	1+,1,XH	Интервальная сумма
SAVEVALUE	11+,1,XH	Общая сумма
TERMINATE		
GENERATE	480	
TERMINATE	1	
START	1	Интервал 1
CLEAR	XH11	
START	1	Интервал 2
CLEAR	XH11	
START	1	Интервал 3
CLEAR	XH11	
START	1	Интервал 4

Карта JOB. Эта карта должна помещаться между последовательными рабочими вычислениями, являющимися частями одной работы. Она выполняет все функции карты CLEAR, а также стирает все описания блоков, функций, переменных, таблиц и памяти. Перед первой работой карта JOB не нужна. Карта JOB возвращает датчик случайных чисел к исходному числу.

Пример 4. Построить программу модели процесса прохождения деталей, поступающих на склад, затем со склада на обработку к одному рабочему со временем 20 ± 12 единиц, распределенным по закону равномерной плотности. Время обработки 19 ± 15 единиц распределено по равномерному закону. Определить коэффициенты занятости рабочего для двух моделей: первая — без ограничения мест на складе, вторая — с ограничением (два места). Во второй модели подсчитывается количество отказов. Промоделировать прохождение 100 деталей в каждой модели. В этом случае программа имеет вид:

	SIMULATE		Начало первой модели
SKL	STORAGE		
	GENERATE	20,12	
	ENTER	SKL	
	SEIZE	OBR	
	LEAVE	SKL	
	ADVANCE	19,15	
	RELEASE	OBR	
	TERMINATE	1	
	START	100	Запуск первой модели
	JOB		Начало второй модели
SKL	STORAGE	2	
	GENERATE	20,12	
	GATE SNF	SKL,FINAL	
	ENTER	SKL	
	SEIZE	OBR	
	LEAVE	SKL	
	ADVANCE	19,5	
	TOLEASE	OBR	
	TERMINATE	1	
FINAL	SAVEVALUE	2+,1,XH	
	TERMINATE	1	
	START	100	Запуск второй модели
	END		

Карта END. Эта карта указывает на конец входной колоды, в которой может содержаться несколько работ. Во входной колоде должна быть только одна карта END. Когда симулятор GPSS доходит до этой карты, он автоматически передает управление супервизору системы.

В языке GPSS разным объектам можно давать одно и то же имя. Например:

```
QUEUE      CHAN
SEIZE      CHAN
```

Возможности языка расширяет косвенная адресация и косвенное описание, в которых используются индексы, применяемые в качестве номера n аргумента блока и в качестве индекса n СЧА, указываемого в некоторой из карт описания. Например: Q10— текущее содержимое очереди 10; S9— текущее содержимое памяти 9; FN2— значение функции 2.

Для записи индекса надо использовать знак «*» и число после него. Индекс определяется значением параметра P*n* транзакта, номер которого записан после *. Например:

SEIZE *6

(т. е. занять устройство, индексный номер которого записан в параметре 6);

Q *3

(т. е. текущее содержимое очереди, индексный номер которой записан в параметре 3).

Косвенная адресация и косвенное описание могут быть использованы совместно и порознь практически для любых аргументов различных типов блоков. Не могут быть описаны косвенно следующие аргументы: модификатор блока ADVANCE; номер параметра; приращение блока INDEX; мнемонические обозначения нижнего и верхнего предела блока PRINT; режим работы (допустимы только стандартные мнемонические обозначения); а также следующие блоки полей В и С в режиме ALL блока TRANSFER; начальный интервал задержки; счетчик числа генерируемых данным блоком транзактов; приоритет; модификатор блока GENERATE.

2.2 ОПИСАНИЕ ОБЪЕКТОВ GPSS

Для описания некоторых характеристик отдельных объектов GPSS (например, емкости памяти, переменных операторов, параметров таблиц) используются карты описания объектов. Формат этих карт подобен формату карт описания блоков.

Язык GPSS оперирует с тремя типами объектов аппаратной категории: устройствами, памятью и логическими ключами.

Устройства. Эти объекты аппаратной категории GPSS применяются для описания таких элементов, которые могут использоваться только одним транзактом в данный момент времени. Каждое устройство предполагает использование четырех СЧА (см. Приложение 3).

Кроме того, каждое устройство имеет шесть стандартных логических атрибутов (СЛА): Un⁵ — истина, если устройство используется, т. е. занимается посредством блока SEIZE или посредством блока PREEMPT; NUn — истина, если устройство не используется ни блоком SEIZE, ни PREEMPT; In⁶ — истина, если устройство занимается с помощью блока PREEMPT; NIn — истина, если устройство не занимается блоком PREEMPT; FVn — истина, если устройство доступно; FNVn — истина, если устройство недоступно.

Память⁷. Эти объекты GPSS описывают такие устройства, которые обслуживают одновременно несколько транзактов. Посредством этого объекта можно представить, например, очередь людей у кассы; рейд, где корабли ожидают освобождения места для разгрузки в порту, и т. д. Каждой памяти присущи семь СЧА (см. Приложение 3).

Емкость каждой памяти определяется в начале процесса испытаний модели с помощью карты STORAGE. Используются два формата — для определения одной или нескольких памятей, Например:

1 STORAGE 237

(в «поле метки» записывается номер памяти, а в «поле операндов» — объем этой памяти).

STORAGE S1,100/S2,200

(«поле метки» пустое, а в «поле операндов» описываются памяти и их емкости). Емкости описываются по следующим правилам: 1) номер памяти обозначается через СЧА Sn; 2) номер памяти отделяется от значения ее емкости запятой; 3) несколько памятей одной и той же емкости могут определяться одним значением, их номера разделяются знаком «—

⁵ U – USE (использовать), n – номер устройства

⁶ I – INVADER (захватить)

⁷ многоканальные устройства (МКУ)

»; SI—S10, 100; 4) отдельные значения емкости разделяются знаком «/».

Каждый объект типа «память» использует шесть СЛИА: SEn — истина, если память пуста, т. е. Sn = 0, Rn = вся емкость; SNEн — истина, если Sn > 0, Rn < вся емкость; SFн — истина, если Sn = вся емкость, Rn = 0; SNFn — истина, если Sn < вся емкость, Rn > 0; SV — истина, если память доступна; SNV — истина, если память недоступна.

Ключи. Эти объекты используются для описания двоичных состояний логического или физического характера. В процессе работы модели один транзакт может установить ключ в определенное положение, другой может изменить его состояние, а третий — использовать это состояние при принятии определенных решений. С этим объектом связаны два СЛИА: LRI⁸ — истина, если ключ *i* выключен, и ложь, если он включен; LSI⁹ — истина, если ключ *i* включен, и ложь, если он выключен.

В вычислительной категории используются объекты трех типов: арифметические переменные, булевские переменные и функции. При построении модели некоторой системы S необходимо выразить арифметические и логические соотношения между атрибутами. Для этого используются арифметические и булевские переменные, которые представляют стандартными числовыми атрибутами V и BV.

Арифметические переменные (определяются перед началом моделирования).

Целые арифметические переменные описываются картой

Метка	Операция	A
j	VARIABLE	выражение

j - имя переменной, используемое для ссылок на неё, обозначается индексом номера или символом. Выражение - арифметическое выражение, включающее в себя различные стандартные числовые атрибуты, а также другие переменные. Результат каждой промежуточной операции в целых переменных преобразуется к целому типу путём отбрасывания дробной части. Для обозначения алгоритмических операторов используются пять символов: «+» — алгебраическое суммирование; «—» — алгебраическое вычитание; «*»¹⁰ — алгебраическое умножение; «/» — алгебраическое деление; @ — деление по модулю. Все указанные действия в выражении осуществляются по тем же правилам и в том же порядке, как, например, в языке ПАСКАЛЬ. Деление на 0 не считается ошибкой (результат при этом равен 0). В выражении не допускаются пробелы (первый пробел означает конец выражения), нет ограничений на количество чисел, но последнее число должно кончатся в колонке 71. Если выражение не помещается на одной карте, то используют несколько карт с дополнительным определением вспомогательных переменных.

Например:

10 VARIABLE Q9+3—P7*FN3

где переменная 10, которая обозначается V10, вычисляется добавлением к длине очереди 9 (Q9) константы 3 и вычитанием результата произведения параметра 7 (P7) текущего транзакта на функцию 3 (FN3).

Действительные переменные:

j	FVARIABLE	выражение
---	-----------	-----------

Действие аналогично VARIABLE, за исключением того, что отдельные элементы и промежуточные результаты не округляются, округляется конечный результат выражения.

Например:

1 FVARIABLE 10(11/3)

1 VARIABLE 10(11/3)

где первая переменная с плавающей запятой равняется 36, т. е. произведение $3,67 \cdot 10 =$

⁸ R –Reset (сбросить)

⁹ S – Set (установить)

¹⁰ В некоторых версиях GPSS символ умножения - #, а не - *

36,7 ограничивается до 36, а вторая переменная равна 30, т. е. 3,67 ограничивается до 3 и умножается на 10.

Булевские переменные. Эти переменные могут быть представлены комбинациями стандартных числовых атрибутов, связанных между собой посредством булевых операторов, включая и другие переменные. Булевские переменные определяются так же, как и арифметические, но вместо арифметических операций проверяются различные логические условия. Существуют три типа операторов: логические, условные и булевские.

- *Логические операторы* используются для определения состояния объектов аппаратной категории и рассматриваются при описании стандартных логических атрибутов «устройств» и «памятей». Например:

3 BVARIABLE FNI2

4 BVARIABLE SF2

где булевская переменная 3 (BV3) равна 1, если устройство 2 не занимается блоком PREEMPT, а булевская переменная 4 равна 1, если память 3 заполнена.

- *Условные операторы* сравнивают алгебраические операнды, которые могут быть константами либо СЧА: G — больше; L — меньше; E—равно; NE—не равно; LE—меньше либо равно; GE—больше либо равно. Например:

11 BVARIABLE V2'G'5

(здесь BV11 равна 1, если $V2 > 5$).

- *Булевские операторы* обозначают следующее: «+»—ИЛИ; «*» — И. Например:

7 BVARIABLE (V2'G'7)*(FNI2+LR7)

(т. е. BV7 равна 1, если $V2 > 7$ при условии, что устройство 2 не занято блоком PREEMPT или логический переключатель LR7 включен).

Функции. Как и переменные, функции не связаны с определенными блоками. Существует пять типов функций:¹¹ 1) непрерывные числовые значения Cn ; 2) дискретные числовые значения Dn ; 3) перечень числовых значений Ln ; 4) дискретные значения атрибутов En ; 5) перечень значений атрибутов Mn . Первые два типа функции, являются основными. На рис. 5 показана непрерывная функция.

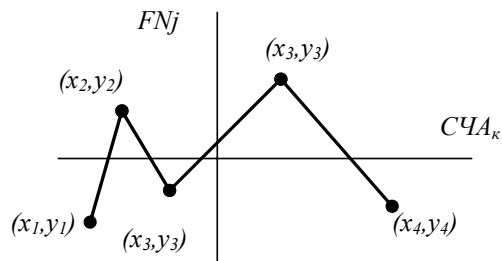


Рис. 5. Непрерывная функция GPSS

Каждая функция определяется посредством карты, имеющей вид

Метка	Операция	A	B
j	FUNCTION	Arg	Cn

В поле A определяется аргумент функции Arg, который может быть СЧА. В поле B указываются тип функции и число точек, для которых вычисляется функция,

Каждая карта FUNCTION сопровождается картами, содержащими значения отдельных точек функции (пары X_i и Y_i). Каждая из этих карт может содержать шесть пар X_i и Y_i , расположенных на 12 шестисимвольных полях. Например, функцию, показанную на рис. 5, можно представить следующим образом:

Метка	Операция

¹¹ n — число точек, задающих функцию

1 FUNCTION RN1,C5
X1,Y1/X2,Y2/X3,Y3/X4,Y4/X5,Y5

Значения X_i и Y_i могут быть нецелыми числами. Значения X должны удовлетворять условию $X_i < X_{i+1}$.

Пользователь может представить координаты функции в свободном формате, а не по шесть на каждой карте. Для этого нужно, чтобы: 1) первое число начиналось с колонки 1; 2) последнее число кончалось до или в 71-й колонке; 3) координаты X_i и Y_i отделялись друг от друга запятой; 4) пары координат отделялись друг от друга знаком «/».

Например, если необходимо, чтобы случайная величина получала значения 1, 4, 5 с относительной частотой 0,40; 0,10; 0,50, то дискретная функция может иметь вид

15 FUNCTION RN8,D3

0.4,1/5,4/1,5

Графическое представление данной функции приведено на рис. 6

У функции аргументом может необязательно RN_j . Например, аргументом может быть любой СЧА. При использовании аргументов, отличных от RN_j , существуют ограничения:

1. Значениями аргумента должны быть только целые числа;
2. Когда значение аргумента выходит за пределы, указанные в описании функции, то значение последней берется равным ближайшим описанным значениям.

Генерация случайных чисел в GPSS имеет следующие особенности. Все случайные равномерно распределенные числа, используемые в GPSS, получаются в результате расчета, который исходит из набора восьми основных чисел, называемых исходными. Пользователь может задать любое из этих чисел (RN_1, \dots, RN_8). В случаях, когда обращение к датчику случайных чисел подразумевается, используется RN_j . Начальные значения и, следовательно, последовательности псевдослучайных чисел, получаемые во всех системных датчиках (RN_1, \dots, RN_8), одинаковы. Для получения различных последовательностей необходимо изменять их начальное значение.

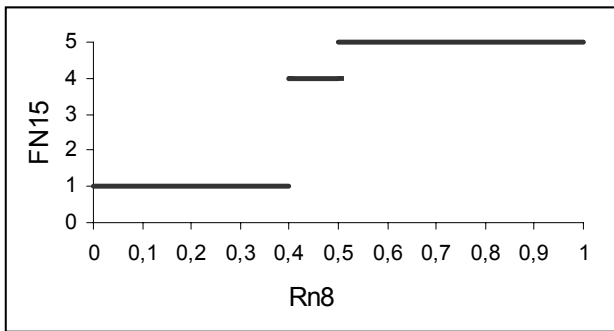


Рис. 6

Моделирование

пуассоновских потоков в GPSS. Наступление событий часто подчиняется следующим условиям:

1. Вероятность того, что наступление события возникает на некотором интервале, пропорционально длине этого интервала.
2. Вероятность того, что возникнут два и более событий в течение

малого промежутка времени, пренебрежимо мала (ординарный процесс).

3. Все интервалы наступления независимы друг от друга (процесс с ограниченным последствием)

Для указанных условий выведена зависимость (закон Пуассона), характеризующая скорость наступления событий

$$P_k(T) = \frac{e^{-\lambda T} (\lambda T)^k}{k!} \quad k = 0, 1, 2, \dots$$

где $P_k(T)$ – вероятность того, что ровно k событий произойдет за время T ; λ – средняя интенсивность наступления событий. Когда интенсивность наступления распределена по закону Пуассона, соответствующие значения интервалов времени наступления

распределены по экспоненциальному закону: $IBH = IBH_{cp} \cdot [-\ln(1-RN_i)]$, где IBH_{cp} – средний интервал времени наступления, RN_i – случайное число ($RN_i \in [0, 1]$). Знак минус принят в выражении потому, что значение логарифма отрицательно. Вычисление натурального логарифма в GPSS не предусмотрено и выражение $-\ln(1-RN_i)$ аппроксимируется ломаной прямой, заданной координатами 24 точек

```
EXPON FUNCTION RN1,C24
0,0/.100,.104/.200,.222/.300,.355/.400,.509/.500,.690
.600,.915/.700,1.200/.750,1.380/.800,1.600/.840,1.830/.880,2.120
.900,2.300/.920,2.520/.940,2.810/.950,2.990/.960,3.200/.970,3.500
.980,3.900/.990,4.600/.995,5.300/.998,6.200/.999,7/1,8
```

График функции EXPON показан на рис. 7.

Например, необходимо смоделировать пуассоновский входящий поток со средним значением 6 заявки в течение каждых 24 часов. Единица измерения – 1 мин. В таком случае блок GENERATE будет иметь вид

```
GENERATE 240,FN$EXPON
```

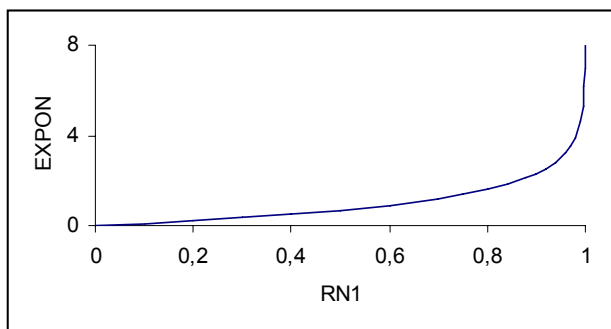


Рис. 7

Операнд А получен путем преобразования интенсивности в соответствующее ей среднее время между последовательными поступлениями заявок: 4 часа или 240 мин. Например, если значение RN_j равно 0,55, значение функции FN\$EXPON будет равно 0,8025. Произведение $0,8025 \cdot 240$ равно 192,6. Следовательно, следующая заявка поступит в модель через 192 мин. Так как, для пуассоновского потока принято предположение, что интервал между двумя последовательными поступлениями не может равен нулю, то рекомендуется значение операнда А блока GENERATE брать не менее 50.

В GPSS предусмотрен сбор и обработка типовых статистических данных по каждому объекту (максимальная и средняя длина очереди, коэффициент загрузки устройства, памяти, среднее время обработки в устройстве и ожидания в очереди и др.), а также дополнительной статистики, заложенной программистом в модели. Процесс имитации функционирования системы во времени (динамика процесса функционирования) может быть представлен и в графическом виде, что особенно эффективно для учебных целей.

В статистической категории GPSS используется два типа объектов: очереди и таблицы.

Очереди. Транзакт помещается в очередь в том случае, когда некоторое устройство не в состоянии обслужить его немедленно (например, устройство занято, либо память переполнена). Статистические данные об очередях могут быть получены в разных точках модели. Для изменения длины очереди в выбранных точках размещается два типа блоков: блок QUEUE и блок DEPART. Блок QUEUE может быть помещен перед любым блоком диаграммы, в котором может возникнуть задержка. При этом транзакты увеличивают длину очереди при входе в блок QUEUE и уменьшают ее при входе в блок DEPART. В

GPSS транзакт может входить в несколько блоков QUEUE одновременно. Каждая очередь имеет семь стандартных числовых атрибутов (см. прил. 3).

Таблицы. Статистические таблицы используются для получения частотных распределений определенных аргументов, которыми могут быть некоторые СЧА (например, времени задержки транзакта в модели в целом или в отдельных ее частях; длин очередей; содержимого памяти и т. д.). У каждой таблицы имеются определенные области значений аргумента. Число попаданий аргумента в каждую из этих областей регистрируется. В конце эксперимента с моделью результаты в таблицах выводятся на печать. С помощью карты TABLE пользователь определяет аргумент и области частотных интервалов. Карта TABLE имеет следующий формат:

Метка	Описание	Табулируемый аргумент	Верхняя граница первого интервала	Размер интервалов	Количество интервалов
		A	B	C	D
j	TABLE	СЧА _i СЧА*n СЧА _i — СЧА*n — RT IA	Kn	Kn	Kn

Знак «—» после СЧА показывает, что используется разностный режим. Табулируемый аргумент RT₁₂ указывает на режим «RT», а IA₁₃ — на режим «IA». В режиме RT добавляется поле E, в котором указывается интервал времени, относительно которого измеряется интенсивность (автомобилей в час, сообщений в сутки и т.д.). Например, если за единицу модельного времени принята одна минута, то карта

TABRT TABLE RT,0,5,10,60

означает, что значение интенсивности указывает на число приходов за 60 минут.

Каждая таблица имеет три СЧА (см. прил. 3): TB_n — средние значения аргумента, которые накапливаются в таблице с последней карты RESET или CLEAR; TD_n — стандартное отклонение, равняющееся корню квадратному из дисперсии аргументов; TC_n — счетчик входов в таблицу, т. е. общее число табулированных аргументов.

Объекты **запоминающей категории** (ячейки и матрицы) обеспечивают обращения к сохраняемым значениям.

Ячейки. Эти объекты используются в модели для сохранения отдельных значений СЧА, чтобы в дальнейшем можно было обращаться к ним. Для этого необходимо резервировать ячейки оперативной памяти ЭВМ, в которых эти значения будут храниться посредством карты SAVEVALUE¹⁴. При принятии определенных решений отдельные транзакты могут обращаться к этим ячейкам и использовать хранящуюся в них информацию. Ячейка SAVEVALUE может сохранять числовое значение в размере формата «слова», «полуслова», «байта», «плавающей точки» в зависимости от ее типа. Ячейкам SAVEVALUE каждого типа соответствует один СЧА (см. прил. 3).

Перед входом в блок SAVEVALUE никакие специальные условия не проверяются, поэтому в этот блок может войти любой транзакт (т. е. он может быть задержан на этом

¹² Rate – интенсивность

¹³ Inter Arrival – интервалы прихода

¹⁴ Save value – сохранить значение

блоке).

Поле А интерпретируется как номер ячейки SAVEVALUE, значение которой изменяется в результате прохождения блока.

На ячейки SAVEVALUE можно ссылаться, можно изменять их содержимое. Для рационального использования оперативной памяти ЭВМ формат ячейки SAVEVALUE должен согласовываться с записываемым значением. Занесение сохраняемых значений осуществляется с помощью блока SAVEVALUE.

В поле В задается значение, которое надо записать в данную ячейку. Ячейки SAVEVALUE могут иметь символические имена. Например:

SAVEVALUE PROCE,PH2,XH

(т. е. значение параметра 2 транзакта, вошедшего в блок SAVEVALUE, записывается в ячейку SAVEVALUE с именем PROCE формата «полуслово»). Или

SAVEVALUE XB4 — 12,10,XH

(где блок SAVEVALUE записывает число 10 в ячейку формата «полуслово», начиная с той, номер которой записан в ячейку формата «байт», и кончая ячейкой 12-го формата «полуслово». Число, записанное в ячейке 4, должно быть меньше 12, иначе заданный диапазон номеров окажется ошибочным).

Формат ячейки SAVEVALUE задается в поле С. Допускаются следующие записи: X, XF, или пробел (ячейка формата «слово»), H или XH (ячейка формата «полуслово»), XB (ячейка формата «байт») и XL (ячейка формата «плавающая точка»). Если записываемое значение превышает максимально допустимое для ячейки SAVEVALUE данного формата, выдается предупреждающее сообщение. Например:

SAVEVALUE HTOT,156,XH

(т. е. в ячейке SAVEVALUE формата «полуслово с именем HTOT записывается число 156);

SAVEVALUE RLCLK,CI,XF

(когда текущее значение условного времени записывается в ячейку SAVEVALUE формата «слово»);

SAVEVALUE QUELN,Q*PB10,XF

(т. е. длина очереди, определяемой параметром 10 формата «байт», записывается в ячейку формата «слово» с именем QUELN).

Содержимое ячеек SAVEVALUE можно изменить заменой добавлением и вычитанием значений. Если за номером ячейки в поле А стоит знак «+», значение, записанное в поле В, добавляется к текущему содержимому ячейки. Знак «—» определяет вычитание значения, записанного в поле В, из текущего содержимого этой ячейки. Отсутствие символа означает замену содержимого. Например:

SAVEVALUE CROSS,V\$RSLT,XF

(т. е. текущее содержимое ячейки GROSS формата «слово» заменяется значением переменной RSLT);

SAVEVALUE GROSS+,V\$RSLT,XF

(в этом случае значение переменной RSLT добавляется к текущему содержимому ячейки GROSS).

Матрицы. Эти объекты дают возможность упорядочить значения, которые нужно сохранить в виде матриц $m \times n$, где m — число строк; n — число столбцов. Пользователь может определить число строк и столбцов, а также размер информации, которая будет храниться (формат «слово», «полуслово», «байт», «плавающая точка») с помощью карты MATRIX. С этим типом объекта связаны СЧА (см. прил.3).

Матрицы. Эти объекты дают возможность упорядочить значения, которые нужно сохранить в виде матриц $m \times n$, где m — число строк; n — число столбцов. Пользователь может определить число строк и столбцов, а также размер информации, которая будет храниться (формат «слово», «полуслово», «байт», «плавающая точка») с помощью карты MATRIX. С этим типом объекта связаны СЧА (см. прил.3). Блоки MSAVEVALUE

присваивают, или изменяют значения элементов матриц.

Карта MATRIX определяет матрицу GPSS.

Метка	Операция	A	B	C
j	MATRIX	A	B	C

j - имя матрицы. A - неиспользуемое поле (для совместимости с прежними версиями GPSS). B - количество строк матрицы. Обязателен. Операнд должен быть константой. C - количество столбцов матрицы. Обязателен. Операнд должен быть константой. Например:
ARRAY MATRIX ,100,5

Этот карта определяет матрицу ARRAY, которая имеет 100 строк и 5 столбцов.

Размер матрицы ограничен -он может содержать не более 8191 элемента. Если есть необходимость в большем количестве элементов нужно определить дополнительную матрицу.

Изменение элементов матриц во время моделирования производится блоком:

MSAVEVALUE	A	B	C	D
------------	---	---	---	---

A - имя или номер матрицы, B - номер строки, C - номер столбца, D - новая величина элемента матрицы.

К группирующей категории относятся два типа объектов: списки и группы.

Списки. Транзакты GPSS хранятся в списках. Существует пять видов списков в одном из которых в любой момент времени может находиться транзакт: 1) текущих событий; 2) будущих событий; 3) пользователя; 4) прерываний; 5) синхронизируемых транзактов.

В списке *текущих событий* транзакты расположены в порядке убывания приоритета; внутри класса транзакты с одинаковыми приоритетами располагаются в порядке поступления их в список. Каждый транзакт в списке текущих событий может находиться либо в активном состоянии (т. е. просматриваться в данный момент условного времени), либо в состоянии задержки.

Если транзакт находится в активном состоянии, то процедура, осуществляющая просмотр, пытается продвигать его к следующим блокам диаграммы. Если движение транзакта блокируется каким-либо объектом аппаратной категории, т. е. вход в следующий блок выполнить невозможно, то он переводится в состояние задержки.

Часть списка текущих событий для транзактов ожидающих изменения состояния объектов аппаратной категории, представляют собой список задержки. Существует четыре вида списков, связанных с устройствами (устройство занято, свободно, обслуживает прерывание, работает без прерываний), пять видов, связанных с памятью (память заполнена, не заполнена, пуста, не пуста, транзакт может войти в память), и два — с логическими ключами (ключ установлен, ключ сброшен).

Таким образом, заблокированный транзакт (например, блоке SEIZE из-за занятости устройства) будет переведен в состояние задержки и помещен в список задержки «устройство занято» для данного устройства. Когда другой транзакт освободит устройство, все транзакты списка задержки «устройство занято» для данного устройства будут переведены в активное состояние, и симулятор GPSS отметит, что произошло изменение состояния устройства. Как только завершается обработка активного транзакта, запускается процедура просмотра, и все транзакты, получившие возможность двигаться, в результате изменения состояния устройства будут продвинуты. То же происходит с любым из перечисленных выше списков задержки в момент, когда соответственно изменяется состояние объекта аппаратной категории.

Пользователю следует учитывать, что для транзактов, заблокированных в блоке TEST или пытающихся выйти из блока TRANSFER (в режимах BOTH, ALL), списков задержки не существует. В этих случаях транзакты остаются в активном состоянии в списке текущих событий и участвуют во всех просмотрах.

Список *будущих событий* содержит транзакты, у которых намеченное время начала движения больше, чем текущее. Такие транзакты размещаются в списке строго в порядке

возрастания времени начала движения, т. е. транзакт, время начала движения которого является ближайшим к значению текущего времени, состоит в списке первым, а транзакт с максимальным временем начала движения — последним. Приоритеты не влияют на порядок транзактов в этом списке.

Список *пользователя* содержит транзакты, удаленные пользователем из списка текущих транзактов с помощью блока LINK.

Список *прерываний* содержит транзакты, обслуживание которых прервано на одном или более устройствах. Транзакты в списке прерываний расположены в порядке поступления. Этот список никогда не обрабатывается как списковая структура; транзакты по мере надобности заносятся в этот список и извлекаются из него. Все транзакты, захватившие устройства (пройдя блок SEIZE) и прерванные другими транзактами (прошедшими блок PREEMPT), помещаются в этот список до тех пор, пока не будут сняты все условия прерываний. Если условия прерывания не снимаются, транзакт может оставаться в списке прерываний неопределенно долго.

Список *синхронизируемых транзактов* содержит транзакты, которые ожидают в блоках ASSEMBLE или GATHER прихода заданного числа транзактов из той же группы: находятся в блоках MATCH в ожидании поступления транзакта той же группы на сопряженный блок MATCH. Этот список также никогда не обрабатывается как списковая структура и представляет собой поднабор набора транзактов.

Группы. В некоторых моделях систем транзакты представляют собой объекты, характеризующиеся общими атрибутами, которые обычно представляются как параметры транзакта и меняются при прохождении транзактов через блок ASSIGN. Однако это дает ограниченный доступ к атрибутам, а иногда приходится менять некоторые атрибуты всех транзактов данной совокупности.

Объекты такого типа предоставляют возможность пользователю обращаться к атрибутам транзактов данной группы. Они работают в одном из режимов: в режиме транзакта или числовом. Режим определяется первым обращением к определенной группе. После того, как режим установлен, его необходимо поддерживать. В режиме транзакта величины, поступающие в группу, представляются номерами по очереди транзактов данной группы. В другом режиме поступающие величины представляются списком числовых значений.

2.3 БЛОКИ, СВЯЗАННЫЕ С ТРАНЗАКТАМИ GPSS

К динамической категории GPSS относятся транзакты, которые в процессе моделирования создаются, размножаются, собираются и уничтожаются. Каждому транзакту соответствует набор параметров, число которых может достигать 1020. Если число параметров не определено, то вводится 12 параметров формата «полуслово» для каждого транзакта. Параметры могут иметь формат: «слово», «полуслово», «байт», «плавающая точка». Транзактам присваивается приоритет от 0 до 127. Если приоритет не назван, то он принимается равным минимальному приоритету, т. е. 0.

С динамической категорией GPSS связаны следующие группы блоков: 1) задержки транзактов по заданному времени; 2) создания и уничтожения транзактов; 3) изменения параметров транзактов; 4) создания копий транзактов; 5) синхронизации движения транзактов.

1. **Группа блоков задержки транзактов по заданному времени.** Из четырех типов событий, которые могут произойти при прохождении транзакта через блок, простейшим является задержка транзактов в течение определенного периода времени. Задать его можно только в блоке ADVANCE. Этот блок задает среднее время выполнения операции в моделируемой системе S, а также разброс времени относительно среднего. В блоке производится определение числа единиц времени, на которое транзакт будет задержан. Это время может выражаться любым целым положительным числом, в том числе и нулем. Если время равно 0, то транзакт в блоке ADVANCE не задерживается и переходит в

следующий блок блок-диаграммы.

Для задания времени пребывания в блоке ADVANCE пользователь указывает среднее время в поле А, а модификатор — в поле В. Если время задержки постоянно, поле В может быть пустым, а если время задержки нулевое, то и поле А может быть пустым. Модификаторы могут быть двух типов: интервал и функция.

Модификатор-интервал используется, когда время задержки транзактов распределено равномерно в некотором заданном диапазоне. Например, если времена распределены равномерно в интервале (10 ± 5) , то среднее—10, интервал—5. В результате с равной вероятностью получаем целые числа от 5 до 15.

Например:

ADVANCE 10,5

(т. е. при использовании модификатора-интервала имеет место задержка на (10 ± 5) единиц времени).

Модификатор-функция используется, если времена задержки транзактов распределены не равномерно, а более сложным образом. При обращении к функции определяется некоторое число — значение функции, и время задержки в блоке определяется умножением среднего на это число (если результат — не целое число, то берется его целая часть).

Например:

ADVANCE 500, FN2

(т. е. при использовании модификатора-функции получается задержка на $500 * FN2$ единиц времени).

2. Группа блоков создания и уничтожения транзактов. В эту группу входят блоки GENERATE, TERMINATE, SPLIT и ASSEMBLE.

Функцией блока GENERATE является создание транзактов, входящих в систему. В поле А задается среднее время между поступлением отдельных транзактов. Как и в блоке ADVANCE, это время может быть модифицировано с помощью модификатора в поле В (интервала или функции). В поле может быть записан «0». Если при вычислении времени появления в системе первого транзакта оно получилось равным 0, то симулятор полагает его равным 1. Среднее время принимается равным 1, если поле В пусто, а в поле А описан модификатор-функция. Задаваемый модификатор-интервал не должен превосходить среднего, записанного в поле А, чтобы не получались отрицательные интервалы между моментами появления транзактов.

Интервал между транзактами, т. е. время появления следующего транзакта, вычисляется только после того, как сгенерированный транзакт покидает блок GENERATE. Поэтому если после GENERATE стоит блок, который может по какой-либо причине задержать сгенерированный транзакт, то время генерации следующего транзакта будет вычислено после снятия блокирующего условия (т. е. когда сгенерированный транзакт пройдет следующий за GENERATE блок). Поэтому средний интервал между транзактами будет больше, чем среднее значение, заданное в поле А, что может привести к ошибке. Избежать ее можно, поместив после GENERATE блок, не задерживающий транзактов.

В поле С записывается начальная задержка. Заданное в этом поле число (без модификации) определяет интервал времени до создания данным блоком первого транзакта. Это число может быть меньше, равно или больше, чем среднее, заданное в поле А.

Поле D задает число транзактов, которое должно быть создано блоком GENERATE. Если это поле пусто, то блок генерирует неограниченное число транзактов.

В поле E задается приоритет (от 0 до 127), присваиваемый генерируемым транзактам. Если в этом поле значение приоритета не задано, генерируемые транзакты имеют нулевой приоритет.

Поля F—1 резервируют для транзакта необходимое число типов параметров.

Максимально допустимое число параметров —1020, т. е. по 255 параметров каждого формата: «слово», «полуслово», «байт», «плавающая точка». Поскольку параметры не кодируются в определенной последовательности, то для определения типа их необходимо использовать индексы. Допустимыми индексами являются: PH — параметр формата «полуслово»; PF — параметр формата «слово»; PB — параметр формата «байт»; PL— параметр формата «плавающая точка». Если поля пусты, то транзакту выделяется 12 параметров формата «полуслово». Если в поле F записан «0», то у транзакта нет параметров. Первоначальные значения параметров равны 0.

Рассмотрим несколько примеров использования блока GENERATE.

GENERATE 10,3,100,16,5,5PB,20PH,3PL,4PF

(транзакты создаются через каждые (10 ± 3) единиц времени, причем первый транзакт создан в момент времени, равный 100, всем 16 транзактам, созданным в этом блоке, присваивается приоритет 5; каждый транзакт имеет по пять параметров формата «байт», по 20 параметров формата «полуслово», по три параметра формата «плавающая точка» и по четыре параметра формата «слово»).

GENERATE 10,2,1000,10,4

(транзакты генерируются через каждые (10 ± 2) единиц времени, причем первый из десяти генерируемых этим блоком транзактов создан в момент времени, равный 1000; всем транзактам присваивается приоритет 4, и все они имеют по 12 параметров формата «полуслово», так как поля, задающие число параметров, пусты).

GENERATE 100,FN\$EXPON,,100

(транзакты создаются через интервал времени, равный произведению 100 на значение функции EXPON; в этом блоке генерируется только 100 транзактов, которые имеют по 12 параметров формата «полуслово», и всем им присвоен приоритет 0).

GENERATE 54,FN\$NORM,,7

(транзакты генерируются через 54 единицы времени, модифицированные функцией NORM, и имеют приоритет 7 и по 12 параметров формата «полуслово»).

GENERATE 10,FN\$EXPON,,,,10PF

(использование модификатора-функции здесь аналогично использованию функций в предыдущих примерах; транзактам присваивается приоритет 0 и все они имеют по десять параметров формата «слово»).

Блок TERMINATE удаляет транзакты из системы. Он используется для изображения окончания пути транзакта в системе (например, в момент прибытия транзакта в пункт назначения). Поле А блока TERMINATE указывает, изменяет ли этот блок содержимое счетчика завершения TG1 в момент поступления транзакта и, если изменяет, то на сколько единиц. В приведенном ниже примере в поле А записано 2. Это значит, что каждый раз, когда транзакт входит в этот блок, из счетчика завершений вычитается 2. Если содержимое этого счетчика задано (специальной картой START) равным 1000 и во всей модели один блок TERMINATE, то до завершения счета через этот блок пройдет 500 транзактов. Если поле не определено, то оно считается равным 0, и транзакты, проходящие через такой блок, не уменьшают содержимого счетчика завершений. Соответственно, модели должен быть хотя бы один блок TERMINATE, у которого поле А не меньше 1, чтобы процесс моделирования закончился. Если в модели такого блока нет, то счет будет продолжаться до тех пор, пока задачу не остановит оператор.

Рассмотрим примеры использования блока TERMINATE:

TERMINATE

(удаляет транзакт, счетчик TG1 не изменяется);

TERMINATE 2

(уменьшается счетчик TG1 на 2).

3. Группа блоков изменения параметров транзактов. Каждый транзакт может иметь от 0 до 1020 параметров, Интерпретация смысла параметров произвольная. В момент

генерации транзакта все его параметры нулевые. Блок ASSIGH¹⁵ является основным средством для задания значений параметров транзактов. В поле А указывается, какой параметр поступившего транзакта должен быть изменен. Следующий непосредственно за номером параметра символ указывает, что нужно сделать с записанным в поле В целым числом: прибавить его к текущему значению параметра, вычесть его из текущего значения параметра; заменить текущее значение этим числом. Эти операции соответственно обозначаются n^+ , n^- , n , где n — номер параметра.

Если в поле С указано какое-либо значение, оно интерпретируется как номер функции. Производится определение значения функции, а результат используется для модификации целого числа, указанного в поле В. Произведение помещается в параметр, указанный в поле А.

В поле D задается тип изменяемого параметра. Допускайте следующие записи: PH, PF, PB, PL, соответствующие параметрам формата: «полуслово» «слово», «байт» и «плавающая точка». Если модификатор-функция не задан, то вместо поля D можно использовать поле С. Например:

ASSIGN 1,475,PH

(т. е. первому параметру формата «полуслово», входящему в блок ASSIGN транзакта, нужно присвоить значение 475).

ASSIGN 12+,45,PB

(т. е. текущему значению 12-го параметра формата «байт» прибавить число 45).

Можно изменять значения нескольких последовательных параметров одного типа следующим образом:

ASSIGN 1—7,5,,PH

(здесь текущее значение параметров с 1-го по 7-й включительно (формата «полуслово») заменяется числом 5).

Если же число 5 следует прибавить к текущему значению этих параметров, то блок следует закодировать так:

ASSIGN 1-7+,5,,PH

Если в поле С задан модификатор-функция и предполагается функция 1, то в поле С должна быть такая запись:

ASSIGN 3,5,1PH

но не такая:

ASSIGN 3,5,FN1,PH

(здесь номер функции определяется вычислением значения функции 1; результат будет одинаковым в том случае, если значение FN1 равно 1).

Пример 5. Построить программу модели процесса прохождения 70 деталей, поступающих с интервалом времени 12 ± 2 единицы и обрабатываемых одним рабочим по пяти последовательно идущим друг за другом операциям, времена выполнения которых распределены в интервале 2 ± 1 единица времени. Распределение всех времени в указанных интервалах равномерно. В результате моделирования требуется определить коэффициент занятости рабочего.

Программа имеет вид:

SIMULATE		Начало моделирования
GENERATE	12,2	Поступление деталей
ASSIGN	2.5,,PB	Счетчик операций
SEIZE	1	Запуск детали
WAIT ADVANCE	2.1	Операция
LOOP	PB2,WAIT	Переход к следующей операции
RELEASE	1	Выпуск детали

¹⁵ ASSIGH - назначить


```
TERMINATE 1
START      70
END
```

4. *Группа блоков создания копий транзактов.* Блок `SPLIT`¹⁶, так же как блок `GENERATE` предназначен для создания транзактов. Но в отличие от `GENERATE` блок `SPLIT` не создает самостоятельных транзактов, а лишь генерирует заданное число копий входящего в него транзакта. Получаемые копии идентичны исходному транзакту. Число копий задается в поле `A`. После прохождения блока `SPLIT` исходный транзакт направляется в следующий блок, а все копии пересылаются по адресу, указанному в поле `B` (в том числе и к следующему блоку). Таким образом, если в поле `A` задано число i , то из блока выйдут $i+1$ транзакт. Далее исходное сообщение и копии являются равноправными и могут проходить снова через любое число блоков `SPLIT`. Все транзакты, полученные копированием, а также копии копий принадлежат к одному ансамблю и далее к этому ансамблю можно применять специальные блоки, осуществляющие обработку ансамблей транзактов (например, `MATCH`, `ASSEMBLE`, `GATHER`).

Получаемый ансамбль транзактов может быть пронумерован. Для этого в поле `C` записывается номер параметра транзакта, в котором будет произведена нумерация. Если в исходном транзакте значение этого параметра было равно k , то после нумерации исходный транзакт получит значение $k+1$, первая копия — $k+2$ и т. д.

Так как копии транзакта могут иметь параметры разных типов, то необходимо использование индекса для указания типа параметра, который берется при объединении в серии. Допускаются индексы: `PH` — параметр формата «полуслово»; `PF` — параметр формата «слово»; `PB` — параметр формата «байт». Исходный транзакт и копии объединяются в серии по заданному параметру. В приведенном ниже примере рассматривается случай, когда параметр 10 формата «байт» имеет значение N при входе транзакта в блок `SPLIT`. При выходе из блока значение его станет $N+1$, а значения параметра 10 (формата «байт») транзактов копий соответственно будут: $N+2$, ...

```
SPLIT 6,NEXTY,10PB
```

Копии, полученные в блоке `SPLIT`, могут иметь число и типы параметров, отличные от исходного транзакта.

В полях `D`, `E`, `F` и `G` блока `SPLIT` определяются число и типы параметров, присваиваемых копиям. Эти поля, начиная с поля `D`, можно задавать в любом порядке. В каждом поле должен быть указан индекс параметра для определения его типа (используются те же индексы, что и в блоке `GENERATE`). В копии переносятся значения только тех параметров исходного транзакта, для которых есть соответствующие номера. Если поля `D`, `F`, `E` и `G` пусты, параметры копий такие же, как и у исходного транзакта. В приведенном ниже примере каждая из восьми копий будет иметь по восемь параметров формата «плавающая точка» и четыре параметра формата «байт». Если исходный транзакт имеет соответствующие параметры, содержимое параметров будет передано копиям:

```
SPLIT 8.NEXTZ,,8PL,4PB
```

Все транзакты, сгенерированные блоками `SPLIT`, из одного исходного транзакта и его копий, принадлежат к одному ансамблю. Число транзактов в ансамбле произвольно. Каждый транзакт, созданный в блоке `GENERATE`, является отдельным ансамблем. Таким образом, число ансамблей в системе произвольно, и ансамбль существует до тех пор, пока в нем находится хоть один транзакт.

5. *Группа блоков синхронизации движения транзактов.* Блок `ASSEMBLE` используется для объединения определенного числа транзактов, являющихся членами одного ансамбля. Число объединяемых ансамблей указывается в поле `A`. Транзакты, принадлежащие одному ансамблю, будут задерживаться в блоке `ASSEMBLE` до тех пор,

¹⁶ Split - расщепить

пока не поступит заданное число транзактов этого ансамбля. В результате на выходе блока появляется один (первый) транзакт ансамбля, а остальные транзакты уничтожаются. В одном блоке ASSEMBLE могут накапливаться транзакты разных ансамблей, транзакты одного ансамбля могут накапливаться в разных блоках ASSEMBLE. Если число собираемых членов ансамбля задается с помощью косвенной адресации, то для его установления используется параметр первого пришедшего транзакта. Рассмотрим примеры использования блока ASSEMBLE:

ASSEMBLE 5

(т. е. собирается пять транзактов, четыре уничтожаются, один переходит в следующий блок);

ASSEMBLE *1

(т. е. собирается число транзактов, равное значению параметра «1» первого транзакта ансамбля).

Пример 6. Построить программу модели процесса прохождения 100 деталей, поступление которых подчиняется равномерному закону в интервале (8 ± 2) единицы времени и обработка производится параллельно двумя рабочими, каждый из которых выполняет свою операцию независимо друг от друга со временем (5 ± 3) единицы, распределенным равномерно. В результате моделирования требуется определить коэффициент занятости рабочих.

Программа имеет вид:

	SIMULATE		Начало моделирования
	GENERATE	8,2	Приход деталей
	SPLIT	1,THIS	Начало обработки детали
	SEIZE	1	Первый рабочий
	ADVANCE	5,3	
	RELEASE	1	
	TRANSFER	,THAT	
THIS	SEIZE	2	Второй рабочий
	ADVANCE	5,3	
	RELEASE	2	
THAT	ASSEMBLE	2	Окончание обработки детали
	TERMINATE	1	
	START	100	
	END		

Действие блока GATHER аналогично действию блока ASSEMBLE. Отличие состоит в том, что после накопления в блоке числа транзактов, указанного в поле A, они все передаются в следующий блок. Блок GATHER позволяет синхронизировать движение транзактов одного ансамбля при их движении по одному пути. Приведем пример использования блока GATHER:

GATHER 3

(т. е. после прихода трех транзактов они поступают в следующий блок).

Пример 7. Построить программу модели процесса прохождения 80 деталей. На производственный участок сборки подшипников поступают обоймы и шарики с интервалом времени (25 ± 4) единицы. На контроль обоймы затрачивается (4 ± 1) единица времени; контроль шариков производится последовательно со временем (2 ± 1) единица времени на шарик. Операция сборки требует одновременного поступления обоймы и всех шариков и производится со временем (4 ± 2) единицы. Все процессы подчиняются равномерному закону. В результате моделирования требуется определить коэффициенты занятости рабочих участка.

Программа имеет вид:

	SIMULATE		Начало моделирования
	GENERATE	25,4	
	SPLIT	8,THAT	Разделение обоймы и шариков
	SEIZE	1	
	ADVANCE	4,1	Контроль обоймы
	RELEASE	1	
	TRANSFER	,FINAL	
THAT	SEIZE	2	
	ADVANCE	2,1	Контроль шариков
	RELEASE	2	
	GATHER	8	
FINAL	ASSEMBLE	9	Комплектация
	SEIZE	3	
	ADVANCE	4,2	Сборка
	RELEASE	3	
	TERMINATE	1	
	START	80	
	END	3	

Блок MATCH предназначен для синхронизации продвижения двух транзактов одного ансамбля, движущихся по разным путям. Для синхронизации необходимы два блока MATCH, находящиеся в соответствующих местах блок-диаграммы и называемые сопряженными. В поле A каждого блока MATCH указывается метка сопряженного ему блока. При подходе транзакта к блоку MATCH проверяется наличие в сопряженном ему блоке транзакта из того же ансамбля. Если в обоих блоках имеются транзакты одного ансамбля, то они одновременно пропускаются через блоки MATCH. Если в сопряженном блоке нет ни одного транзакта данного ансамбля, то поступивший транзакт будет ожидать поступления транзакта того же ансамбля в сопряженный блок MATCH, после чего они оба будут пропущены в следующие за блоками MATCH блоки.

Одна и та же пара блоков MATCH может одновременно синхронизировать любое число пар транзактов из разных ансамблей. Транзакты одного ансамбля также могут синхронизировать в любом числе пар блоков MATCH. Следует отметить, что блок MATCH может быть сопряжен сам себе. При этом его действие будет аналогично действию блока GATHER с параметром 2 в поле A. Рассмотрим соответствующие примеры использования блока MATCH

```
AAA MATCH BBB
```

```
.....
```

```
BBB MATCH AAA
```

(в этом случае транзакт AAA будет ожидать прихода члена того же ансамбля в блок MATCH с меткой BBB);

```
CCC MATCH CCC
```

(здесь транзакт будет ждать прихода члена того же ансамбля в этот же блок MATCH).

Пример 2.8. Построить программу модели прохождения 500 деталей. Детали поступают с интервалом времени (300 ± 50) единиц. Обработку производят двое рабочих, которые выполняют по две операции. После первой операции, выполняемой первым рабочим со временем (70 ± 20) единиц и вторым со временем (60 ± 30) единиц, производится операция сверки, время выполнения которой принимается равным нулю. После сверки выполняется вторая операция первым рабочим со временем (20 ± 10) единиц и вторым — (50 ± 20) единиц. Затем третий рабочий производит сборку изделия из этих

деталей со временем (50 ± 20) единиц. Все процессы подчиняются равномерному закону. В результате моделирования необходимо определить коэффициенты занятости рабочих.

Программа имеет вид:

	SIMULATE		Начало моделирования
	GENERATE	300,50	Поступление деталей
MANA	SEIZE	1	
	ADVANCE	70,20	Первый рабочий, первая операция
HERE	MATCH	THERE	Сверка
	ADVANCE	20,10	Первый рабочий, вторая операция
	RELEASE	1	
	TRANSFER	,MANC	
MANB	SEIZE	2	
	ADVANCE	60,30	Второй рабочий, первая операция
THERE	MATCH	HERE	Сверка
	ADVANCE	30,20	Второй рабочий, вторая операция
	RELEASE	2	
MANC	ASSEMBLE	2	
	SEIZE	3	
	ADVANCE	50,20	Сборка
	RELEASE	3	
	TERMINATE 1		
	START	500	
	END		

2.4 БЛОКИ, ОПИСЫВАЮЩИЕ РАБОТУ ОБОРУДОВАНИЯ

Язык GPSS оперирует с тремя типами объектов, принадлежащих аппаратной категории: устройствами, памятью и ключами.

Устройство терминологии GPSS является аналогом обслуживающего прибора системы массового обслуживания (Q-схемы) [18]. В любой момент времени устройство может быть занято только одним транзактом. Состояние устройства меняют шесть блоков: SEIZE, RELEASE, PREEMPT, RETURN, FUNAVAIL, FAVAIL, которые используются попарно.

В результате входа транзакта в блок SEIZE устройство, указанное в этом блоке, будет занято. Оно останется занятым, пока тот же транзакт не пройдет соответствующий блок RELEASE. Если какой-либо транзакт занимает устройство, описанное в поле A блока SEIZE, то никакой другой транзакт не сможет войти в этот блок (и вообще не сможет захватить это устройство в любом другом блоке SEIZE). Транзакт может занять любое число устройств.

Блок RELEASE служит для освобождения устройства, которое ранее было захвачено проходившим в блоке SEIZE транзактом. При выполнении этого блока программы задержка возникнуть не может. Устройство освобождается в момент входа транзакта в блок RELEASE. Освобождение выполняется только тем транзактом, которым оно было занято. Если перед SEIZE задерживаются несколько транзактов, они обслуживаются в соответствии с правилом: «первым пришел — первым обслужен».

Рассмотрим пример использования блоков SEIZE и RELEASE:

```
SEIZE      1
ADVANCE   10,5
RELEASE   1
```

(т. е. происходит занятие устройства I, задержка от 5 до 15 единиц времени и освобождение устройства I).

Блок PREEMPT фиксирует использование устройства на более высоком уровне, чем блок SEIZE, а также приостанавливает обслуживание транзакта, захватившего устройства ранее, и дает возможность прерванному транзакту захватить устройство после того, как закончится обслуживание прервавшего транзакта.

Если при выполнении блока PREEMPT оказывается, что одно прерывание уже произошло (устройство обслуживает прерывание), то блок не выполняется и соответствующий транзакт задерживается до тех пор, пока не освободится устройство. Затем обслуживается новый прерывающий транзакт» а не прерванный. Исключением из описанных выше правил является случай, когда блок PREEMPT работает в режиме приоритетов, т. е. в поле В стоит PR. При этом в действиях блока PREEMPT предусмотрен случай разрешения прерывания на основании результата анализа приоритетов транзактов.

Для последующей обработки прерванных транзактов существуют следующие возможности:

- в поле С может быть описан какой-либо блок, на который будет передан прерванный транзакт. При этом прерванный транзакт продолжает претендовать на данное устройство;
- если прерванный транзакт находится в блоке ADVANCE, то вычисляется остаток времени (от момента прерывания до момента выхода из блока ADVANCE) и полученное значение помещается в параметре, описанном в поле D блока PREEMPT. В этом случае прерванный транзакт будет послан в блок, указанный в поле С блока PREEMPT. Прерванный транзакт продолжает претендовать на данное устройство;
- если в поле Е блока PREEMPT стоит запись RE, то будут производиться обычные операции, за исключением того, что прерванный транзакт не участвует больше в конфликте из-за захвата устройства.

Блок RETURN сигнализирует об окончании прерывания. При входе в блок задержка возникнуть не может, но закончить прерывание может только тот транзакт, который перед этим прошел блок PREEMPT, относящийся именно к данному устройству. Прерывание заканчивается в момент входа в блок RETURN. Время, в течение которого транзакт находится в прерванном состоянии, не фиксируется. На прерывания имеются следующие ограничения: нельзя производить прерывание транзакта, захватившего или прервавшего обслуживание других транзактов более чем на 255 устройствах одновременно, задержка прерванного транзакта начинается с момента первого прерывания и оканчивается в момент окончания последнего. Неважно, занято или свободно устройство, для которого выполняется блок PREEMPT.

Рассмотрим пример использования блоков PREEMPT и RETURN:

```
PREEMPT 16  
ADVANCE 150  
RETURN 16
```

(т. е. реализуется прерывание на устройстве 16, задержка на 150 единиц, а затем возврат устройства).

Блок FUNAVAIL¹⁷ выполняет операции, переводящие устройство в состояние недоступности. Недоступность устройства предупреждает занятие или прерывание устройства последующими сообщениями. При этом возможно задание специальных режимов работы блока FUNAVAIL, обеспечивающих окончание обслуживания последнего транзакта, передачу его на обслуживание к другому блоку, дообслуживание транзакта после окончания периода недоступности. Номер или диапазон номеров устройств, переводимых в состояние недоступности, записывается в поле А. Поля В...Н предназначены для задания специальных режимов.

¹⁷ F - FACILITIES (устройства), UNAVAIL (сделать недоступным)

Блок FAVAIL¹⁸ делает доступными устройства, указанные как номер или диапазон номеров устройств в поле А. Блок FAVAIL отменяет все режимы, заданные блоком FUNAVAIL для данных устройств.

Рассмотрим пример использования блоков FUNAVAIL и FAVAIL:

```
FUNAVAIL 1—15
ADVANCE 30
FAVAIL    1—10
ADVANCE 15
FAVAIL    11—15
```

(т. е. недоступные устройства с 1-го по 15-е становятся доступными: через 30 единиц времени — устройства с 1-го по 10-е, а через 45 единиц времени — устройства с 11-го по 15-е).

Часто два и более приборов работают рядом. GPSS позволяет моделировать однородные параллельные приборы с помощью специальных средств, называемых *многоканальным устройством*, или *памятями*. Для введения в программу памяти требуется описать ее картой STORAGE, а изменение состояния памяти производится блоками ENTER, LEAVE, SUNAVAIL, SAVAIL. Блок ENTER моделирует событие «занятие одного из группы параллельно работающих приборов», LEAVE¹⁹ — «освобождение параллельно работающего прибора».

Поле А блока ENTER интерпретируется как номер памяти, поле В — указывает число единиц памяти, занимаемых транзактом при входе в блок. При выходе транзактов из блока ENTER никаких изменений в содержимом памяти не происходит.

Если поле В пустое, то число единиц памяти полагает равным 1. В этом поле может быть записан 0 и тогда содержимое памяти увеличится на 0 единиц. Если в памяти нет достаточного числа свободных единиц, чтобы удовлетворить запрос транзакта, то этот транзакт не может быть обслужен блоком ENTER. Если для последующего транзакта число единиц памяти достаточно, он входит в память раньше первого.

Поле А определяет номер памяти, а поле В — число единиц, которые надлежит освободить при входе транзакта в блок LEAVE. Не всегда освобождается такое же число единиц памяти, как было занято. Транзакт, освобождающий память, не обязательно должен был ее занимать. Однако необходимо, чтобы в сумме освобождалось столько единиц памяти, сколько было занято (иначе содержимое памяти будет возрастать и после ее заполнения в системе наступит насыщение), и чтобы содержимое памяти не стало отрицательным (т. е. не освобождалось больше единиц памяти, чем было занято). При выполнении блока LEAVE задержки не возникают. Аналогично тому, как в блоке ENTER транзакт может занять 0 единиц памяти, в блоке LEAVE он может освободить 0 единиц.

Пример 8. Построить программу модели многоканальной системы массового обслуживания с очередью. В парикмахерской три мастера, очередь к ним общая. Клиенты приходят с интервалом (10 ± 5) минут, обслуживание парикмахером клиента занимает (25 ± 10) минут. Смоделировать работу парикмахерской в течение одного рабочего дня (8 часов).

	SIMULATE		Начало моделирования
PARIK	STORAGE	3	Три парикмахера
	GENERATE	10,5	Приход клиентов
	QUEUE	ОСНЕР	Занятие очереди
	ENTER	PARIK,1	Начало стрижки
	DEPART	ОСНЕР	Освобождение очереди
	ADVANCE	25,10	Стрижка

¹⁸ AVAIL (сделать доступным)

¹⁹ LEAVE - выйти

LEAVE	PARIK,1	Завершение стрижки
TERMINATE		Уход клиента
GENERATE	480	Время завершения моделирования
TERMINATE	1	
START	1	Системный счетчик равен 1

Схема данной СМО аналогична схеме на рис. 3. Отметим разницу между одноканальным и многоканальным устройствами. Одноканальное устройство может заниматься или захватываться транзактом, многоканальное устройство – только заниматься.

В следующем примере сама очередь представлена в виде многоканального устройства.

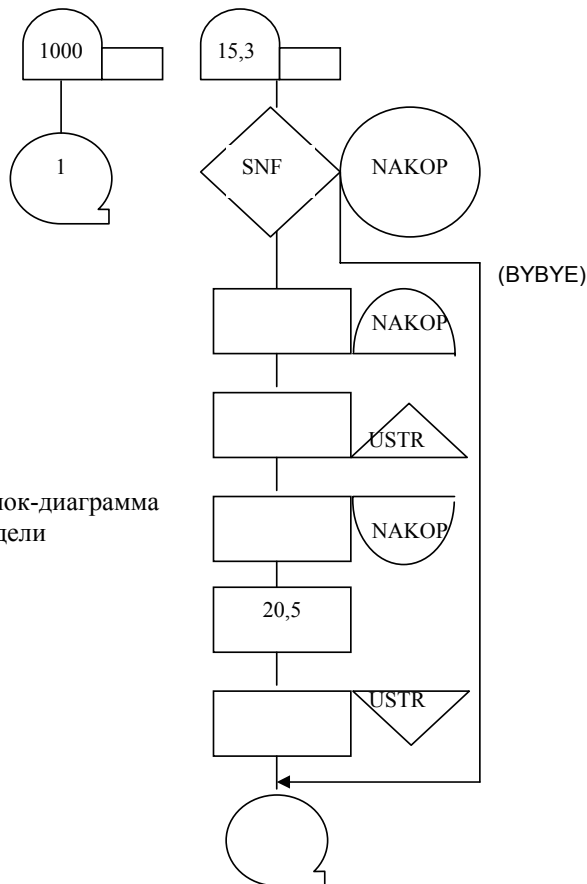


Рис. 8. Блок-диаграмма GPSS-модели

Пример 9. СМО состоит из одного прибора и очереди перед ним. Обработка заявки в приборе занимает (20 ± 5) единиц времени. Очередь ограничена длиной 4. Заявки приходят каждые (15 ± 3) единиц времени и если в очереди нет свободных мест, то заявки покидают модель необслуженными. Промоделировать систему в течение 1000 единиц времени.

SIMULATE		Начало моделирования
NAKOP STORAGE	4	Длина очереди – 4 заявки
GENERATE	15,3	Приход заявок
GATE SNF	NAKOP, BYBYE	Пропустить заявку на

*нижеследующий блок, если очередь NAKOP заполнена не полностью,

*в противном случае заявку отправить на метку BYBYE

ENTER	NAKOP	Поступление в очередь
SEIZE	USTR	Занятие устройства

	LEAVE	NAKOP	Освобождение очереди
	ADVANCE	20,5	Задержка
	RELEASE	USTR	Освобождение устройства
BYBYE	TERMINATE		Уход заявки
	GENERATE	1000	Время завершения моделирования
	TERMINATE	1	
	START	1	Системный счетчик равен 1

Блок-диаграмма GPSS-модели показана на рис. 8.

Блок SUNAVAIL²⁰ переводит накопитель в состояние недоступности, при котором транзакты не могут войти в накопитель. Уменьшение содержимого накопителя в этот период может происходить путем прохождения транзактами блока LEAVE. Номер или диапазон номеров накопителей, переводимых в состояние недоступности, записывается в поле A.

Блок SAVAIL переводит заданный накопитель из состояния недоступности в состояние доступности. Если данный накопитель уже доступен, то блок SAVAIL никаких действий не выполняет. Номер или номера накопителей, переводимых в состояние доступности, записываются в поле A.

Рассмотрим пример использования блоков SUNAVAIL и SAVAIL:

```
SUNAVAIL  2—5
TEST      NE      PHI,O,NEXT
SAVAIL    2—5
```

(т. е. недоступные накопители с 1-го по 5-й становятся доступными при появлении транзакта с ненулевым параметром I типа «полуслово»).

Логические ключи в GPSS предназначены для описания элементов моделируемой системы S, которые могут находиться только в двух состояниях. Статистика о работе ключей не собирается. Логические ключи не имеют СЧА, а только два логических атрибута, принимающие значения 0— при невыполнении и 1— при выполнении следующих условий: LR²¹ — ключ в состоянии «0»; LS²² — ключ в состоянии «1».

В начале моделирования ключ может быть установлен в состояние «1» с помощью карты INITIAL, и изменение состояния ключа производится в процессе моделирования блоком LOGIC.

Блок LOGIC используется для установки логических ключей, состояние которых может быть запрошено в любом другом месте модели. При входе в блок LOGIC задержки не возникает. Состояние логического объекта, указанного в поле A, изменяется одним из трех способов: может быть установлен S, сброшен R или инвертирован I. Вид изменения определяется соответствующим мнемоническим обозначением, помещаемым непосредственно за блоком LOGIC.

Рассмотрим пример применения блока LOGIC:

```
LOGIC S  41
LOGIC R  165
LOGIC I  4
```

(т. е. установить ключ 41, сбросить ключ 165, инвертировать ключ 4).

2.5 ИЗМЕНЕНИЕ МАРШРУТОВ ТРАНЗАКТОВ

В GPSS поток транзактов обычно проходит блоки последовательно, поэтому предполагается, что, выполнив блок с номером n , транзакт переходит к выполнению блока с номером $n + 1$. Однако в моделируемой реальной системе S в зависимости от

²⁰ S – STORAGE (память)

²¹ L – LOGIC (логический), R – RESET (сбросить)

²² S – SET (установить)

создавшихся условий поток транзактов может быть задержан или направление его движения изменено. Для моделирования таких ситуаций на пути потока транзактов может быть помещен блок специального типа, который реализует переход в зависимости от создавшихся условий. В зависимости от способа и условий, сложившихся в модели в данный момент, транзакт по указанию пользователя может быть задержан или направлен в блок с любым указанным номером. Например, перед входом транзакта в блоки некоторых типов (например, блок SEIZE) симулятор GPSS проверяет состояние устройства, указанного в этом блоке. Транзакт войдет в блок, когда соответствующее устройство свободно. Если транзакт не может перейти к выполнению этого блока, то симулятор предоставляет две возможности: либо транзакт может быть задержан до тех пор, пока не освободится требуемое устройство, либо направлен по другому пути. В группу изменения маршрутов сообщений входят блоки GATE, TEST, TRANSFER.

Блок GATE²³. Этот блок используется для определения состояния объектов устройств без изменения их состояния. Блок GATE работает в двух режимах: 1) отказа или условного входа;

2) перехода или безусловного входа.

При работе в режиме отказа блок GATE не пропускает транзакты, если соответствующий объект не находится в требуемом состоянии. Если же поставленное в блоке условие удовлетворяется, блок разрешает вход транзактов.

Если в поле В указано наименование (номер) блока, то вместо отказа блок GATE будет посылать транзакт на указанный блок. Таким образом, если поле В пустое, блок работает в режиме отказа, если не пустое — в режиме перехода.

Существует шесть условий или логических атрибутов, описывающих состояние устройств, памяти, ключей и условия синхронизации. Мнемонические обозначения проверяемого условия записываются непосредственно после GATE. Поле А определяет номер объекта аппаратной категории (устройства, памяти или ключа). Состояние устройства описывается следующими условиями: FNU²⁴ — устройство не используется, свободно; FU — устройство используется, занято (обслуживает захвативший транзакт или прерывание); FNI — устройство работает без прерывания (свободно или обслуживает захвативший его транзакт); FI — устройство обслуживает прерывание; FV — устройство доступно; FNV — устройство недоступно.

Состояние памяти описывается следующими условиями: SE — память пуста; SNE — память не пуста; SF — память заполнена; SNF — память не заполнена; SV — память доступна; SNV — память недоступна.

Состояние ключа описывается двумя условиями: LR — логический ключ в состоянии «выключен»; LS — логический ключ в состоянии «включен».

Следующие два мнемонические обозначения подробно описаны ниже: M — блок GATE проверяет выполнение условия синхронизации в указанном блоке блок-диаграммы; NM — блок GATE проверяет невыполнение условия синхронизации в указанном блоке блок-диаграммы.

Рассмотрим пример использования блока GATE:

режим отказа	
GATE SF	167
GATE LS	265
GATE FU	19

(т. е. выполняются следующие действия: блокировать транзакт до тех пор, пока память 167 не будет заполнена; блокировать транзакт до тех пор, пока ключ 265 не установлен, блокировать транзакт до тех пор, пока устройство 19 не освободится);

²³ GATE (впустить)

²⁴ F – FACILITIES (устройства), N – NOT (не), U – USE (использовать)

режим перехода

GATE FI 34,ALTR

(т. е. если устройство 34 прервано, то перейти к ALTR).

Блок TEST. Блок TEST описывает условие, которое проверяет при входе в него транзакта и определяет направление дальнейшего движения транзакта в зависимости от этого условия, которое записывается в виде алгебраического соотношения двух аргументов. При выполнении соотношения транзакт пропускается в следующий за блоком TEST блок. В случае невыполнения соотношений транзакт направляется в блок, метка которого указана в поле С. Если поле С пусто, то транзакт блокируется блоком до выполнения соотношения.

Проверяемое соотношение записывается в поле операций сразу за блоком TEST. При этом можно использовать следующие мнемонические обозначения операций сравнения: L — «меньше чем»; LE — «меньше или равно»; E — «равно»; NE — «не равно»; G — «больше чем»; GE — «больше или равно».

Соотношение рассматривается между первым • и вторым аргументами, записываемыми в полях А и В соответственно. Аргументы должны принадлежать к стандартным числовым атрибутам.

Если поле С не пусто, то транзакт всегда может войти в блок TEST и в зависимости от соотношения аргументов будет передан либо в следующий блок, либо в блок, указанный в поле С. Если поле С пусто, то транзакт при невыполнении условия не сможет войти в блок TEST и симулятор в каждый момент времени будет проверять, не изменилось ли блокирующее условие. Такой режим является нежелательным вследствие больших затрат машинного времени на многократные проверки блокирующего условия. Например:

```
TEST L      S1,K10
TEST NE     PF1,PF2
TEST GE     Q2,FN1,СЕН
```

Блок TRANSFER. Этот блок обычно используется для того, чтобы передать входящие в него транзакты в блоки, не следующие по номеру за ним. Передача может быть выполнена логически, статистически, условно и безусловно. Вид передачи определяется мнемоническим обозначением, указанным в поле А блока TRANSFER. При безусловной передаче указывается один следующий блок (путь), а при условной передаче—два или больше путей. Выбор того или иного пути определяется режимом работы блока TRANSFER. Поле В определяет первый (или единственный) из следующих блоков. Если следующий блок поля В описан на следующей карте, то поле В может быть оставлено пустым. Когда симулятор ПМДС встречает карту — описание блока TRANSFER с пустым полем В, то он записывает в поле В этой карты номер следующего блока.

Поле С определяет следующий блок и интерпретируется в соответствии с режимом работы блока TRANSFER. Существуют различные режимы работы блока TRANSFER.

Если поле А блока TRANSFER пусто, то все транзакты, поступающие на этот блок, будут передаваться в блок, номер которого указан в поле В. Например:

```
TRANSFER ,NEXT
```

Если в поле А блока TRANSFER стоит BOTH, то каждый транзакт, поступающий на этот блок, проверяет два пути. Сначала проверяется блок, указанный в поле В, и если транзакт не может войти в этот блок, он пытается войти в блок, указанный в поле С. Если транзакт не может войти и в этот блок, он задерживается в блоке TRANSFER и повторяет опрос блоков, указанных в полях В и С, при каждом изменении текущего времени в модели, пока не находит выхода. Этот режим выбора существенно увеличивает время работы, так как если оба выхода заблокированы, опрос их состояния производится очень часто. Например:

```
TRANSFER BOTH,,SAM
```

SEIZE	52
SAM SEIZE	84

(транзакт блока TRANSFER пытается войти в блок SEIZE; если соответствующее устройство занято, то он направляется в блок SAM; если транзакт не может войти в блок SAM, то он остается в данном блоке TRANSFER и повторяет описанные выше операции при каждом изменении текущего времени).

Если в поле А стоит ALL, то транзакты, входящие в блок, могут опрашивать много путей (блоков). Поле В определяет первый выбираемый блок, поле С — последний. Поле D определяет индексную константу i . Эта константа дает пользователю возможность опрашивать определенные блоки, находящиеся между теми, которые заданы в полях В и С. Транзакт, вошедший в блок TRANSFER, далее пытается выйти через блок, указанный в поле В. Если блок, указанный в поле В, занят, то транзакт пытается войти в блоки с номерами $Y + i, Y + 2i, \dots, Z$, где Y — номер блока, указанного в поле В, Z — номер блока, указанного в поле С. При этом $Z - Y + M_i$ где M — некоторое целое положительное число. Если транзакт не может войти ни в один из этих блоков, он остается в блоке TRANSFER и при изменении текущего времени повторяет все описанные выше операции. Если задан режим ALL, пользователь должен проследить за тем, чтобы номера просматриваемых блоков совпадали с номерами, присваиваемыми блокам симулятором GPSS. Например:

```

TRANSFER ALL,FIRST,LAST,3
FIRST SEIZE 1
      ASSIGN 12,K1,,PB
      TRANSFER ,LAST+2
      SEIZE 2
      ASSIGN 12,K2,,PB
      TRANSFER ,LAST+2
LAST SEIZE 3

```

(транзакт пытается войти в блок FIRST; если он получает отказ, что определяется типом блока FIRST, то транзакт пытается войти в блок FIRST+3; если и в этом блоке будет отказ, то транзакт пытается войти в блок FIRST+6 (LAST); если транзакт не смог войти ни в один из трех блоков, он остается в блоке TRANSFER и программа (система) переходит к обработке другого транзакта). При каждой попытке продвинуть транзакты, находящиеся в блоке TRANSFER, все проверки, перечисленные выше, производятся сначала, как и в режиме BOTH, поэтому время счета также может значительно возрасти.

Если в поле А стоит SIM, то выбирается один из двух возможных путей. Каждый входящий в блок транзакт содержит информацию для принятия решения. В каждом транзакте есть индикатор SIM, установленный в 0 или 1. Если значение индикатора SIM равно 1, то транзакт будет послан в блок указанный в поле С, а индикатор SIM будет установлен в 0. Если индикатор SIM равен 0, то транзакт может направляться по выбранному для него пути. Если транзакт получает отказ при попытке войти в какой-либо блок, этот индикатор устанавливается в 1, фиксируя тот факт, что некоторое условие не выполняется. Если одновременно должны удовлетворяться несколько условий, каждый из них может быть проверен, и если все они удовлетворяются, то индикатор SIM транзакта остается равным 0. Если не выполняется хотя бы одно из условий, индикатор SIM устанавливается в 1. В таких случаях блок, заданный в поле С, указывает блок, в котором проверялось первое условие, и транзакт выполняет проверку условий, пока все они не будут удовлетворяться одновременно. Блок ADVANCE также устанавливает индикатор SIM в 0. Проверка состояний связана с блоками, которые могут задерживать транзакты. Для проверки условий используется блок GATE, описанный ранее.

Существует режим проверки одновременного выполнения нескольких условий. В приведенном ниже примере транзакты не могут перейти к блоку SEIZE, если устройства 1, 2 и 15 не будут одновременно свободны. При входе транзактов в блок TRANSFER

проверяется значение индикатора SIM этого транзакта. Если транзакт был задержан в каком-либо из блоков GATE, его индикатор установлен в 1. Когда индикатор проверяется в блоке TRANSFER, оказывается, что транзакт был задержан, и он отсылается к тому блоку, где производилась первая проверка (CPU), и вся последовательность проверок повторяется. При проверке (т. е. при прохождении блока TRANSFER) с индикатором SIM, индикатор всегда сбрасывается в 0, поэтому если транзакт повторяет последовательность проверок в тот же момент времени, т. е. без задержки, то после второй попытки он пройдет, так как индикатор SIM устанавливается в 0 при первой попытке, Например:

CPU	GATE NU	1
	GATE NU	2
	CATE NU	15
	TRANSFER	SIM,,CPU
	SEIZE	1,5

Рассмотрим статистический режим выбора. Если в поле A блока TRANSFER записана десятичная дробь, то производится случайный выбор между блоками, описанными в поле B и в поле C. Вероятность выбора блока C задается дробью. Например:

TRANSFER, 370,THIS,THAT

(т. е. режим выбора задан числом 370, при этом 37% всех транзактов, поступивших на блок TRANSFER, будет передаваться на блок поля C, а 63% — на блок поля B).

Для каждого отдельного транзакта, естественно, выбирается только один из двух путей. Выбор производится для каждого транзакта только один раз, в момент входа в блок TRANSFER. Кроме этого, пользователю предоставлены еще четыре режима выбора: PICK, SBR, FN и P.

2.6 БЛОКИ ДЛЯ СБОРА СТАТИСТИКИ

В статистической категории GPSS используется два типа объектов: очереди и таблицы. Очереди вводятся в моделирующую программу для регистрации статистической информации о процессе ожидания. Постановка транзакта в очередь производится блоком QUEUE, а удаление транзакта из очереди — блоком DEPART. Таблицы вводятся в моделирующую программу для сбора статистических данных. Ввод производится с помощью карты TABLE, а регистрация статистических данных в таблице — при входе транзакта в блок TABULATE.

Блок QUEUE. Этот блок аналогичен блоку ENTER и извещает симулятор GPSS о том, что в данной точке блок-диаграммы нужно собирать статистику об очереди. Номер очереди, в которую пользователь хочет занести транзакт, задается в поле A блока QUEUE. При записи нового транзакта в очередь определяется длина интервала времени, в течение которого длина очереди оставалась неизменной. При входе транзакта в блок QUEUE текущая длина очереди увеличивается на число единиц, указанное в поле B, после чего симулятор сравнивает новую длину очереди с максимальной, достигнутой до сих пор. Если новое значение больше максимального, то старое максимальное значение очереди заменяется на новое. Кроме того, счетчик общего числа единиц, прошедших через очередь, увеличивается на то же число единиц. Общее число единиц используется для определения взвешенного среднего времени пребывания транзакта в очереди.

Блок DEPART. Этот блок аналогичен блоку LEAVE. Поле A интерпретируется как номер очереди. В поле B задается число единиц, на которое уменьшается длина очереди. При входе транзактов в блок DEPART число единиц, указанное в поле B, вычитается из длины очереди. Как и в блоке LBAVE, если поле B пусто, то его значение принимается равным 1.

Симулятор вычисляет длину интервала времени, в течение которого транзакт находился в очереди, и, если длина интервала получилась равной 0, указанное в поле B число единиц добавляется к счетчику, регистрирующему число транзактов, прошедших

через блок без задержки.

Можно заносить в таблицу время пребывания транзакта в очереди. В примере, приведенном ниже, время пребывания в очереди 10 будет заноситься в таблицу QTABLE 15. В ней будет 100 интервалов: 0, 5, 10 ...490.

Рассмотрим пример использования блоков QUEUE и DEPART

```
QUEUE      10
SEIZE      1
DEPART     10
ADVANCE    150, 5
RELEASE    1
QTABLE     10, 0, 5, 100
```

Блок TABULATE. Пользователю GPSS позволено для сбора нужных статистических данных вводить таблицы нескольких типов. Информация заносится в таблицы с помощью специального блока TABULATE в момент входа транзакта в этот блок. Пользователь должен задать ряд характеристик для каждой из таблиц, которую он хочет использовать. Характеристики таблицы задаются специальной картой описания таблицы - TABLE (QTABLE).

При входе транзакта в блок TABULATE симулятор ПМДС записывает в соответствующую таблицу определенную статистическую информацию. Поле А определяет номер этой таблицы, в которой накапливается соответствующая статистическая информация. Вид заносимой информации определяется на специальной карте описания таблицы помимо способа обработки. Эта карта определяет размер, а также число и границы интервалов таблицы. В каждом блоке TABULATE может быть задано число единиц, добавляемых к числу наблюдений того интервала, в который попадает при данном обращении аргумент. Это число единиц задается в поле В. Если поле В пусто, то это число принимается равным 1. Например:

```
TABULATE 10
```

(т. е. в интервалы таблицы 10 добавляется 1);

```
TABULATE 48, 2
```

(т. е. в интервалы таблицы 48 добавляется по 2 единицы).

В GPSS предусмотрено несколько специальных режимов табулирования, которые указываются в поле А карты описания таблицы. Знак «—» за величиной, записанной в поле А, указывает на то, что в таблицу заносится не само значение, а разность между текущим значением этой величины и последним значением, занесенным в таблицу. Этот режим называется *разностным*. Первое обращение к таблице симулятором автоматически исключается.

В качестве примера приведен случай разностного режима табулирования:

```
16      TABLE 10-, 1, A50
```

Таблица TABLE 16 является взвешенной, поэтому в поле D должен стоять любой буквенный символ и за ним — число классов частоты.

Если в поле А стоит мнемоническое обозначение RT, то при входе транзакта в блок TABULATE, который связан с таблицей, заполняемой таким способом, обращение к классам частот не производится. Вместо этого число единиц, заданное в блоке TABULATE, добавляется к счетчику числа входов в таблицу. Карта описания таблицы должна содержать в поле В временной интервал. Если этот интервал равен, например, 1000, значение счетчика числа входов будет заноситься в таблицу по истечении каждых 1000 единиц времени. После занесения счетчик сбрасывается в 0. Таким образом, при этом способе определяется распределение числа «заявок», поступающих за 1000 единиц времени.

Рассмотрим пример использования режима RT:

```
TABULATE 19
19 TABLE RT,0,10,75,1000
```

Если в поле А стоит мнемоническое обозначение IA, то при входе транзакта в блок TABULATE, который соответствует таблице, заполняемой таким способом, симулятор ПМДС определяет время, прошедшее с момента последнего обращения к этой таблице. Полученное значение заносится в таблицу. Как и в случае разностного режима, первое обращение к таблице исключается симулятором автоматически. Такая таблица представляет собой распределение промежутков времени между моментами поступления транзактов в данную точку программы.

Приведем пример использования режима IA:

```
TABULATE 26
26 TABLE IA,0,10,100
```

2.7 ОСНОВНЫЕ ВЫВОДЫ ПО ЯЗЫКУ GPSS

Язык GPSS представляет собой интерпретирующую языковую систему, применяющуюся для описания пространственного движения объектов. Такие динамические объекты в GPSS называются транзактами и представляют собой элементы потока. В процессе имитации транзакты "создаются" и "уничтожаются". Функцию каждого из них можно представить как движение через модель М с поочередным воздействием на ее блоки. Каждый транзакт имеет набор параметров. В GPSS также введены понятия прибора, многоканального устройства (накопителя) и очереди, которые соответствуют постоянным элементам системы.

1. Прибор - это элемент системы, который может находиться в двух состояниях "свободно" и "занято". В состоянии "занято" прибор переводится транзактом, поступившим в него. Все остальные транзакты, после этого поступившие к прибору, встают в очередь к нему, т. е. включаются в упорядоченный список, и ожидают освобождения прибора. Занявший прибор транзакт через некоторое время освобождает его (переводит в состояние "свободно"). Прибор занимается транзактом, стоящим первым в очереди, либо, если очередь пуста, остается в состоянии "свободно" до прихода следующего транзакта.

2. Многоканальное устройство - это элемент системы, имеющий определенную емкость, которая измеряется целым положительным числом. Транзакт занимает какое-то количество каналов и, через некоторое время, освобождает их. Причем число освобождаемых каналов может отличаться от первоначально занятых. Как и в приборах, организуется очередь, если нет требуемого количества каналов.

3. Очередь - это элемент системы, упорядоченный список транзактов к прибору или многоканальному устройству.

Приборы, многоканальные устройства и очереди идентифицируются числовыми номерами или символьными именами. Под символьным именем в GPSS понимается последовательность от 3 до 5 символов, причем первые три символа обязательно латинские буквы.

Модель на языке GPSS строится путем объединения набора стандартных блоков в некоторую программу, определяющую логику функционирования исследуемой системы.

В языке имеется более 40 блоков, каждый из которых имеет определенное стилизованное обозначение и наименование, указывающее на функциональное назначение блока. Для детализации функций, выполняемых блоком, у каждого из них

имеется набор параметров.

Каждому транзакту в исследуемой системе в соответствие ставится сегмент модели, т. е. самостоятельная последовательность связанных блоков, которая описывает весь жизненный цикл временных элементов одного класса, начиная с их появления в системе и заканчивая их уходом из нее. Модель может включать несколько сегментов.

Каждый сегмент модели начинается с одного или нескольких блоков GENERATE, которые создают транзакты, и заканчивается блоком TERMINATE, который эти транзакты удаляет из модели. Каждый транзакт, появившийся в модели, последовательно обрабатывается блоками, входящими в сегмент. Одновременно в модели обрабатывается только один транзакт, остальные транзакты модели ждут своей очереди на обработку.

Порядок обработки транзактов задается календарем модели, который в GPSS называется цепью будущих событий (ЦБС). ЦБС - это упорядоченный по времени начала обработки список транзактов, обработка которых запланирована на будущие моменты времени. Те транзакты, которые могут обрабатываться в данный момент системного времени, но из-за сложившихся в системе условий не обрабатываются, включаются в другой упорядоченный список-цепь текущих событий (ЦТС). Работа монитора моделирующей программы на языке GPSS заключается в том, что в начале из ЦБС выбирается транзакт с наименьшим временем начала обработки и переносится в ЦТС. Время начала обработки этого транзакта присваивается таймеру модели. Затем монитор последовательно пытается начать обработку всех транзактов, которые находятся в ЦТС. Если какой-то транзакт начинает обрабатываться, то он обрабатывается до тех пор, пока либо выйдет из модели, либо его обработка будет задержана на какое-то время и транзакт будет перенесен в ЦБС, или очередной по описанию сегмента блок не сможет обрабатывать транзакт в связи со сложившимися в модели условиями.

Транзакты в ЦТС просматриваются до тех пор, пока ни один из них дальше обрабатываться уже не сможет. Тогда монитор опять выбирает из ЦБС транзакт с минимальным временем, изменяет значение таймера и все выше описанное повторяется. Модель заканчивает работу тогда, когда переменная модели, называемая "счетчик завершения", получит значение меньше или равное 0. Начальное значение счетчика завершения задается при запуске модели в карте START, а изменяется блоком TERMINATE.

ЛИТЕРАТУРА

1. Армстронг Дж. Моделирование цифровых систем на языке VHDL. 1992
2. Бусленко Н.П. Моделирование сложных систем. М.: Наука, 1978
3. Васильев В.И., Романов Л.Г., Червоный А.А. Основы теории систем: конспект лекций. М.: МГТУ ГА, 1994
4. Васильев В.И., Иванюк А.И., Свириденко В.А. Моделирование систем гражданской авиации. М.: Транспорт, 1988
5. Веников В.А., Веников Г.В. Теория подобия и моделирования. М.: Высш. шк., 1984
6. Гнеденко Б.В., Коваленко И.Н. Введение в теорию массового обслуживания. М.: Наука, 1987
7. Ермаков С.М., Мелас В.Б. Математический эксперимент с моделями сложных стохастических систем. 1993
8. Калашников В.В., Рачев С.Т. Математические методы построения стохастических моделей обслуживания. М.: Наука, 1988
9. Киндлер Е. Языки моделирования. М.: Энергоатомиздат, 1985
10. Клейнен Дж. Статистические методы в имитационном моделировании. [в 2-х вып.] М.: Статистика, 1978
11. Математическое моделирование: Методы, описания и исследования сложных систем: [Сб. ст.] / под ред. Самарского А.А. М.: Наука, 1989
12. Питерсон Дж. Теория сетей Петри и моделирование систем. М.: Мир, 1984
13. Полляк Ю.Г., Филимонов В.А. Статистическое машинное моделирование средств связи. М.: Радио и связь, 1988
14. Прицкер А. Введение в имитационное моделирование и язык СЛАМ II. М.: Мир, 1987
15. Романцев В.В., Яковлев С.А. Моделирование систем массового обслуживания. 1995
16. Сабинин Д.Ю. Статистическое моделирование технических систем. 1993
17. Советов Б.Я., Яковлев С.А. Моделирование систем: Курсовое проектирование. М.: Высш. Шк., 1988
18. Советов Б.Я., Яковлев С.А. Моделирование систем. - М.: Высшая школа, 1995
19. Технология системного моделирования / под ред. С.В.Емельянова. М.: Машиностроение, 1988
20. Четвериков В.Н., Баканович Э.А. Стохастические вычислительные устройства систем моделирования. М.: Машиностроение, 1989
21. Шеннон Р.Ю. Имитационное моделирование систем - искусство и наука. М.: Машиностроение, 1989
22. Шрайбер Т. Дж. Моделирование на GPSS.

Примечание: В респ. библиотеке имеется, № по каталогу 3844 А837 1183439 3844.15

Примечание: В респ. библиотеке имеется, № по каталогу 381 В924 340850их 3817

Примечание: В респ. библиотеке имеется, № по каталогу 058 В191 1101445 058

Примечание: В респ. библиотеке имеется, № по каталогу 327 В292 640358

Примечание: В респ. библиотеке имеется, № по каталогу В18 Г561 1041525

Примечание: В респ. библиотеке имеется, № по каталогу В18 К17 1078211 В183.53

Примечание: В респ. библиотеке имеется, № по каталогу 381 К412 992773

Примечание: В респ. библиотеке имеется, № по каталогу В18 К48 719784кх В181

Примечание: В респ. библиотеке имеется, № по каталогу 3973 М34 1115460 3973.233-018

Примечание: В респ. библиотеке имеется, № по каталогу В18 П352 952264 В181

Примечание: В респ. библиотеке имеется, № по каталогу 388 П1515 1086255

Примечание: В респ. библиотеке имеется, № по каталогу 3973 П774 1049762 3973.2-108

Примечание: В респ. библиотеке имеется, № по каталогу У9(2)21 С56 1077029 У9(2)212.53

Примечание: В респ. библиотеке имеется, № по каталогу 381 Т384 1098746 3817

Примечание: В респ. библиотеке имеется, № по каталогу 3965 4-522 1111460 3965.7

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ 1. Условные обозначения на блок-диаграммах GPSS

Pri111

ПРИЛОЖЕНИЕ 2.

Основные сокращения и обозначения симулятора GPSS

TRANS	Номер транзакта
BDT	Время выхода сообщения из блока (это либо значение абсолютного условного времени, при котором транзакт покинет блок ADVANCE , либо значение абсолютного условного времени в момент, когда сообщение вышло из последнего блока ADVANCE , либо нуль, если сообщение еще не входило ни в один блок ADVANCE)
BLOCK	Номер блока, в котором сообщение находится в данный момент времени;
PR	Уровень приоритета сообщения (0..127);
SF	Режим выбора (определяет режим выбора следующего блока: <ul style="list-style-type: none"> • пробел означает переход к следующему блоку; • "A" означает режим ALL блока TRANSFER); • "B" означает, что транзакт находится в блоке TRANSFER, работающем в режиме BOTH;
NBA	адрес следующего блока, в который должен войти транзакт;
SET	номер следующего транзакта того же семейства (при создании транзакта в SET записывается номер самого транзакта. При входе в блок SPLIT в SET записывается номер следующего транзакта из образуемого семейства);
MARK	отметка времени (записывается в момент создания транзакта или при входе транзакта в блок MARK с пустым полем A);
P1, ..., P8	текущие значения параметров 1..8;
S1	индикатор просмотра (при S1 = 1 симулятор не будет пытаться продвинуть транзакт до измерения блокирующих условий; при снятии блокирующего условия S1 устанавливается равным нулю для всех транзактов, задержанных данным условием);
T1	индикатор трассировки (устанавливается в единицу блоком TRACE и в нуль блоком UNTRACE);
D1	индикатор задержки для блока TRANSFER SIM ;
C1	индикатор списка <ul style="list-style-type: none"> • C1 = 0 - транзакт в списке пользователя, • C1 = 1 - движение транзакта прервано, • C1 = 2 - транзакт в списке текущих событий, • C1 = 4 - транзакт в списке будущих событий;
MC	индикатор синхронизации (если MC = 4, то транзакт находится в блоках MATCH , ASSEMBLER , или GATHER);
PC	счетчик прерываний;
PF	флаг прерываний.

ПРИЛОЖЕНИЕ 3.
Стандартные числовые атрибуты (СЧА) GPSS

Тип	СЧ А	Определение	Блоки	Примечание
1	2	3	4	5
Транзакт (TRANSACTIONS)	PS	Текущее значение параметра формата полуслово или слово	ASSIGN, INDEX, INCREMENT, MARK, LOCATE, USING, DECREMENT, LOOP SPLIT	Первоначальное значение приоритета назначается в GENERATE, диапазон 0-127 MP\$ном, где номер параметра транзакта - целое число 0-100 (по умолчанию)
	PRS1	Приоритет	PRIORITY	
	MS1	Транзитное время	MARK, ASSIGN (при условии, что в операнде В СЧА ACS1, CS1)	
	MPS	Параметрическое транзактное время		
	XNS1	Номер транзакта		
Блоки (BLOCKS)	NS	Счетчик входов в блок		Значения СЧА собираются автоматически
	WS	Счетчик текущего содержимого блока		
Переменные (VARIABLES)	VS	Арифметическая и логическая переменная		
Функции (FUNCTIONS)	FNS	Функции		
Таблицы (TABLES)	TBS	Среднее значение аргумента таблицы	TABULATE	Значения СЧА собираются автоматически
	TCS	Счетчик входов в таблицу		
	TDS	Среднее квадратическое отклонение аргумента		
Ячейки (SAVEVALUES)	XHS	Текущее содержимое ячейки (формата полуслово-XHS, слово-XFS, XS)	SAVEVALUE, SINGREMENT, SDECREMENT	Первоначальное значение ячеек при инициализации модели 0 или задается картой INITIAL
	XS			
	XFS			
Матрицы сохраняемых величин (MSAVEVALUES)	MHS(a, b)	Элемент строки "a", столбца "b". Полуслово.	MSAVEVALUE INITIAL	
	MXS(a, b)	Элемент строки "a", столбца "b". Полное слово.		

Тип	СЧ А	Определение	Блоки	Примечание
1	2	3	4	5
Списки пользователя (USERS CHAINS)	CAS	Среднее число элементов в списке	LINK, UNLINK	Значения СЧА собираются автоматически
	CHS	Текущее число элементов в списке		
	CMS	Максимальное число элементов в списке		
	CCS	Общее число входов		
Ключи (LOGICS)	LRS	Ключ сброшен (логическое 1) или ключ установлен (логическое 0)	LOGIC	Первоначальное состояние ключа при инициализации модели «сброшен» или «установлен», если использовать карту LINITIAL
Очереди (QUEUES)	QS	Текущая длина очереди	QUEUE	Значения СЧА собираются автоматически
	QAS	Средняя длина очереди		
	QMS	Максимальная длина очереди		
	QCS	Общее число входов		
	QZS	Количество нулевых входов		
	QTS	Среднее время пребывания транзакта в очереди		
	QXS	Среднее время пребывания транзакта в очереди без учета нулевых входов		
Памяти (STORAGES)	SS	Текущее содержимое памяти	ENTER, LEAVE	Значения СЧА собираются автоматически
	RS	Число свободных единиц памяти		
	SRS	Коэффициент использования		
	SAS	Среднее содержимое памяти		
	SMS	Максимальное содержимое памяти		
	SCS	Общее число входов		
	STS	Среднее время занятости единицы памяти		
	SES	Память пуста (логические 0 или 1)		
	SNES	Память не пуста (логические 0 или 1)		
	SFS	Память заполнена		

Тип	СЧ А	Определение	Блоки	Примечание	
1	2	3 (логические 0 или 1)	4	5	
	SNFS	Память не заполнена (логические 0 или 1)			
Приборы (FACILITES)	FS	Состояние устройства (логические 0 или 1)	SEIZE RELEASE PREEMPT RETURN	Состояние устройств устанавливается автоматически	
	FIS	Прибор находится в состоянии прерывания (логические 0 или 1)			
	FNIS	Прибор не находится в состоянии прерывания (логические 0 или 1)			
	FNUS	Прибор не используется (логические 0 или 1)			
	FSS	Номер транзакта, занимающего прибор			
	FPS	Номер прерывающего транзакта			
	FRS	Коэффициент использования устройства			
	FCS	Общее число входов			
	FTS	Среднее время пребывания транзакта в устройстве			
Системные атрибуты (SYSTEM ATTRIBU-TES)	ACS1	Абсолютное время	RESET	Относительное время после RESET обнуляется	
	CS1	Относительное время			
	TGS1	Содержимое счетчика завершения	START TERMINATE		Содержимое счетчика уменьшается на число единиц, указанных в блоке TERMINATE
	RNSj	Случайное число в интервале [0, 1], и используемое как аргумент функции, и случайное число в интервале [0, 999] в карте f[variable]			

ПРИЛОЖЕНИЕ 4.
Блоки GPSS

Название блока	Операнд						
	A	B	C	D	E	F	G
ADVANCE	к, СЧА*N время задержки	к, СЧА*N модификатор A					
ASSEMBLE	к, СЧА*N количество транзактов						
ASSIGN	к, СЧА*N [+, -] [номер парам.]	к, СЧА*N					
BUFFER	Нет операндов						
DECREMENT [вычесть]	к, СЧА*N Номер параметра	к, СЧА*N значение					
DEPART							
ENTER	к, СЧА*N Номер памяти	к, СЧА*N Число Ячеек					
GATHER	к, СЧА*N Число транзактов						
CATE_aux где aux: SE,SF,SNE,SNF - LR,LS- NI,NU,U,I- M,MN-	к, СЧА*N [номер: памяти, ключа, прибора, блока]	Метка, к, СЧА*N					
GENERATE	к, СЧА*N интервал времени	к, СЧА*N модификатор A	к, СЧА*N время входа 1- транзакта	к, СЧА*N число транзактов к,	СЧА*N приоритет	к, СЧА*N число параметров	к, СЧА*N тип параметра
INCREMENT [сложить]	к, СЧА*N номер параметра	к, СЧА*N значение					
INDEX	к, СЧА*N номер параметра	к, СЧА*N значение					
LEAVE	к, СЧА*N номер памяти	к, СЧА*N число ячеек (по умолчанию 1)					
LINK	к, СЧА*N	LIFO, P*N FIFO	метка, номер блока (альтернативный выход)				

Название блока	Операнд						
	A	B	C	D	E	F	G
LOCATE	к, СЧА*N номер транзакта	тип (X, XB, XF, XH, P, PH, PF, PB)	к, СЧА*N номер параметра или ячейки	метка, альтернативный выход			
LOGIC_аих где аих: S - установить I – инвертировать R - сбросить	к, СЧА*N номер ключа						
LOOP	к, СЧА*N номер параметра	метка (выход при [р*N] = 0)					
MARK	к, СЧА*N номер параметра						
MATCH	к, СЧА*N (номер блока)						
PREEMPT	к, СЧА*N (номер прибора)						
PRINT	СЧА1	СЧА2	СЧ А3	СЧ А4	СЧ А5	СЧ А6	СЧ А7
PRIORITY	к, СЧА*N приоритет	BUFFER					
QUEUE	к, СЧА*N [номер очереди]	к, СЧА*N [количество освобождающихся единиц, по умолчанию 1]					
RELEASE	к, СЧА*N [номер прибора]						
RETURN	к, СЧА*N [номер прибора]						
SAVEVALUE	к, СЧА*N [+, -] [номер прибора]	к, СЧА сохраняемое значение	тип ячейки [H, F, XH, XF по умолч. F]				

Название блока	Операнд						
	A	B	C	D	E	F	G
SDECREMENT [вычесть]	к, СЧА*N [номер ячейки]	к, СЧА*N значение	тип ячейки [H, F, XH, XF по умолчанию F]				
SEIZE	к, СЧА*N [номер прибора]						
SINCREMENT [сложить]	к, СЧА*N [номер ячейки]	к, СЧА*N значение	тип ячейки [H, F, XH, XF по умолч. F]				
SPLIT	к, СЧА*N [число копий]	Метка [блок вывода]	к, СЧА*N [номер параметра]	к, СЧА*N [число параметров]			
TERMINATE	к число транзактов (по умолчанию 0)						
TEST_aux где aux: E, NE, LE, G, GE	к, СЧА*N	к, СЧА*N Метка альтернативного блока					
TRACE	Без операндов						
TRANSFER	FN	Номер функции					
	P BOTH, СЧА*N, k SBR	Номер параметра Метка блока Номер параметра, метка блока	М етка блока М етка блока				

Название блока	Операнд						
	A	B	C	D	E	F	G
UNLINK	ALL, _	Номер или метка блока	Номер или метка блока	Номер или метка блока	Значение шага k		
	PICK	Номер или метка блока	Номер или метка блока	Номер или метка блока			
USING	k, СЧА*N номер цепи пользователя	метка блока	метка блока	число транзактов	число параметров	число параметров	альтернативный выход
UNTRACE	Нет операндов						

ПРИЛОЖЕНИЕ 5.
Карты GPSS

Название карты	Операнд						
	A	B	C	D	E	F	G
END	нет операндов						
EQU	k	(-32766< k <32766)					
DQTABLE	k номер очереди	k верхняя граница 1 интервала	k величина интервала	k число интервалов			
DTABLE	k , СЧА*N аргумент	k верхняя граница 1 интервала	k величина интервала	k число интервалов			
FUNCTION	СЧА*N аргумент	тип функции [C,D]					
FVARIABLE	Логические операции: см. VARIABLE , арифметические операции: + сложить, - вычесть, / разделить, (), * умножить, @ сложить по модулю						
INITIAL	Номер ячейки [XH,XF,X]	k ,...					
JOBTAPE	Имя файла	метка					
LINITIAL	LSSN1	LSSN2	LSSN3	NSSN4	L SSN5	L SSN6	L SSN7
REALLOCATE	Мнемониче ское обозначение	k , ...		(повторное описание)			
RESET	нет операндов						
REWIND	k количество файлов						
RMULT	RNS1	RNS2	RNS3	RNS4	R NS5	R NS6	R NS7
TABLE	СЧА*N [-], IA , аргумент RT	k верхняя граница 1 интервала	величина интервала k	число интервалов k			
VARIABLE	Логические операции: = (равно), < (меньше), > (больше), <= (меньше или равно), >= (больше или равно), & (и), \ (или); арифметические операции: см. FVARIABLE						
QTABLE	Номер очереди [-]	k верхняя граница 1 интервала	размер интервала k	число интервалов k			
START	k счетчик транзактов	отмена печати NP	k интервал выдачи печати	1- выдача сообщений о тразактах			
STORAGE	S*N	k значение	, разделитель				
SIMULATE	нет операндов						

Название карты	Операнд						
	A	B	C	D	E	F	G
WDQTABLE	номер очереди k	k верхняя граница интервала 1	размер интервала k	число интервалов k			
WDTABLE	номер очереди k	k верхняя граница интервала 1	размер интервала k	число интервалов k			

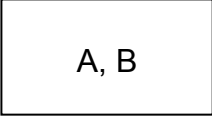
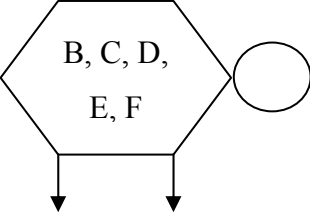
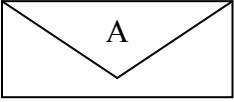
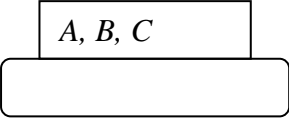
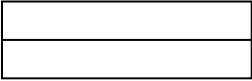

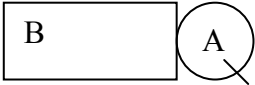
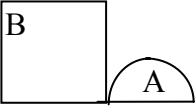
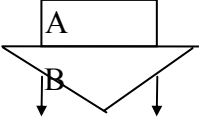
ПРИМЕЧАНИЕ: N может принимать значения:

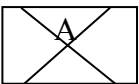
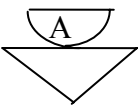
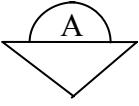
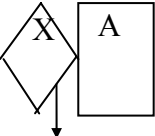
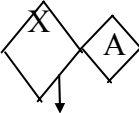
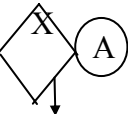
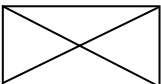
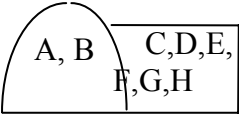
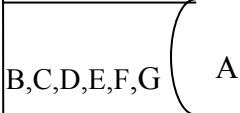
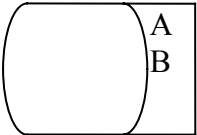
1. ***num**, где **num** - номер параметра транзакта
2. **CЧА\$к**, где **к** - константа

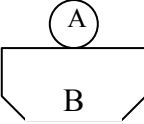
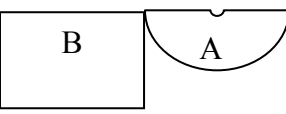
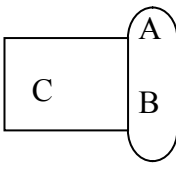
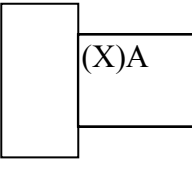
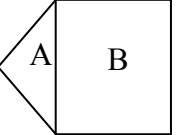

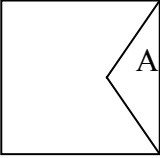

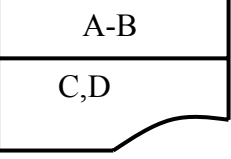
ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ 1

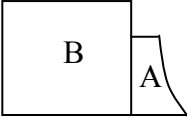
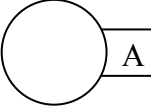

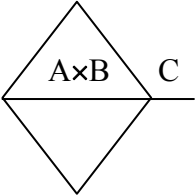
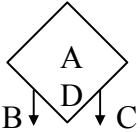
Условные обозначения на блок-диаграммах GPSS

Имя блока	Обозначение блока	Назначение блока
ADVANCE		Задерживает транзакт на время $A \pm B$, если $B = \text{const}$, или $A \cdot B$, если $B = \text{функция}$
ALTER		Изменяет атрибуты членов группы A
ASSEMBLE		Собирает A транзактов одного ансамбля, пропускает в следующий блок первый транзакт, остальные уничтожает
ASSIGN		Присваивает параметру A входящего транзакта значение B , модифицированное значением C
BUFFER		Прерывает обработку транзакта и возвращает симулятор к началу списка текущих событий
CHANGE		Заменяет в процессе счета блок с номером A на блок с номером B
DEPART		Обеспечивает освобождение в очереди A B единиц
ENTER		Обеспечивает вхождение транзакта в накопитель A с занятием B единиц памяти
EXAMINE		Изменяет маршрут движения в зависимости от состояния членов группы A

Имя блока	Обозначение блока	Назначение блока
EXECUTE		Выполняет операции блока с номером А.
FAVAIL		Объявляет группу устройств А доступным
FUNAVAIL		Объявляет группу устройств А недоступным
CATE (X) LS LR		Проверяет условие нахождения логического ключа А в состоянии X
GATE (X) I NI U NU		Проверяет условие нахождения устройства А в состоянии X
GATE (X) SE SF SNE SNE		Проверяет условие нахождения накопителя А в состоянии X
GATHER		Собирает А транзактов одного ансамбля и пропускает их одновременно в следующий блок
GENERATE		Генерирует транзакты через А единиц времени, модифицированных В с задержкой С, D транзактов, с приоритетом Е, форматом F
HELP		Объединяет модули ПМДС с модулями на языках Ассемблера, Паскаля и т.д.
INDEX		Записывает сумму значений параметра А и величины В в параметр 1 типа А

Имя блока	Обозначение блока	Назначение блока
JOIN		Включает в группу A транзакт или числовое значение B
LEAVE		Освобождает в памяти A B единиц памяти
LINK		Удаляет транзакт из списка текущих событий и помещает в список пользователя
LOGIC	<div style="display: flex; align-items: center;"> <div style="writing-mode: vertical-rl; transform: rotate(180deg); margin-right: 10px;">(X) S R I</div>  </div>	Устанавливает логический ключ A в состояние X
LOOP		Осуществляет повторение A раз группы блоков от адреса B до данного блока
MARK		Осуществляет отметку времени в параметре A
MATCH		Синхронизирует движение транзактов по блок-диаграмме совместно с блоком MATCH с меткой A
PREEMPT		Выполняет приоритетную обработку в устройстве A
PRINT		Осуществляет вывод статистики C в пределах от A до B

Имя блока	Обозначение блока	Назначение блока
PRIORITY		Присваивает входящему транзакту приоритет А
QUEUE		Обеспечивает занятие В мест в очереди А
RELEASE		Освобождает устройство с номером А
RETURN		Снимает прерывание с устройства А
SAVAIL		Объявляет группу памятей А доступными
SAVEVALUE		Сохраняет заданное значение В в ячейке А
SCAN		Обрабатывает атрибуты членов группы А
SEIZE		Занимает устройство с номером А
SPLIT		Генерирует А копий входящего транзакта и направляет их по адресу В; основной транзакт переходит в следующий блок
SUNAVAIL		Объявляет группу памятей А недоступными

Имя блока	Обозначение блока	Назначение блока
TABULATE		Табулирует значения входящих транзактов в таблице А
TERMINATE		Уничтожает А транзактов
TRACE		Прослеживает движение транзактов
TEST X NEGATIVE		Проверяет соотношение x между А и В и направляет входящий транзакт в следующий блок при выполнении или по адресу С при невыполнении соотношения
TRANSFER		Изменяет направление движения транзактов согласно режиму А

Основные сокращения и обозначения симулятора GPSS

TRANS	- Номер транзакта
BDT	- Время выхода сообщения из блока (это либо значение абсолютного условного времени, при котором транзакт покинет блок ADVANCE , либо значение абсолютного условного времени в момент, когда сообщение вышло из последнего блока ADVANCE , либо нуль, если сообщение еще не входило ни в один блок ADVANCE)
BLOCK	- Номер блока, в котором сообщение находится в данный момент времени;
PR	- Уровень приоритета сообщения (0..127);
SF	- Режим выбора (определяет режим выбора следующего блока: <ul style="list-style-type: none"> • пробел означает переход к следующему блоку; • "A" означает режим ALL блока TRANSFER); • "B" означает, что транзакт находится в блоке TRANSFER, работающем в режиме BOTH;
NBA	- адрес следующего блока, в который должен войти транзакт;
SET	- номер следующего транзакта того же семейства (при создании транзакта в SET записывается номер самого транзакта. При входе в блок SPLIT в SET записывается номер следующего транзакта из образуемого семейства);
MARK	- отметка времени (записывается в момент создания транзакта или при входе транзакта в блок MARK с пустым полем A);
P1, ..., P8	- текущие значения параметров 1..8;
S1	- индикатор просмотра (при S1 = 1 симулятор не будет пытаться продвинуть транзакт до измерения блокирующих условий; при снятии блокирующего условия S1 устанавливается равным нулю для всех транзактов, задержанных данным условием);
T1	- индикатор трассировки (устанавливается в единицу блоком TRACE и в нуль блоком UNTRACE);
D1	- индикатор задержки для блока TRANSFER SIM ;
C1	- индикатор списка <ul style="list-style-type: none"> • C1 = 0 - транзакт в списке пользователя, • C1 = 1 - движение транзакта прервано,

- **C1 = 2** - транзакт в списке текущих событий,
- **C1 = 4** - транзакт в списке будущих событий;

MC - индикатор синхронизации (если **MC = 4**, то транзакт находится в блоках **MATCH, ASSEMBLER**, или **GATHER**);

PC - счетчик прерываний;

PF - флаг прерываний.

ПРИЛОЖЕНИЕ 3

Стандартные числовые атрибуты (СЧА) GPSS

Тип	СЧА	Определение	Блоки	Примечание
1	2	3	4	5
Транзакт (TRANSACTIONS)	PS	Текущее значение параметра формата полуслово или слово	ASSIGN, INDEX, INCREMENT, MARK, LOCATE, USING, DECREMENT, LOOP SPLIT	Первоначальное значение приоритета назначается в GENERATE , диапазон 0-127 MP\$ном , где номер параметра транзакта - целое число 0-100 (по умолчанию)
	PRS1	Приоритет	PRIORITY	
	MS1 MPS	Транзитное время Параметрическое транзактное время	MARK, ASSIGN (при условии, что в операнде В СЧА AC\$1, CS1)	
	XNS1	Номер транзакта		
Блоки (BLOCKS)	NS	Счетчик входов в блок		Значения СЧА собираются автоматически
	WS	Счетчик текущего содержимого блока		
Переменные (VARIABLES)	VS	Арифметическая и логическая переменная		
Функции (FUNCTIONS)	FNS	Функции		
Таблицы (TABLES)	TBS	Среднее значение аргумента таблицы	TABULATE	Значения СЧА собираются автоматически
	TCS	Счетчик входов в таблицу		
	TDS	Среднее квадратическое отклонение аргумента		
Ячейки	XHS	Текущее	SAVEVALUE,	Первоначальное значение ячеек

Тип	СЧА	Определение	Блоки	Примечание
1	2	3	4	5
(SAVEVALUES)	X\$ XF\$	содержимое ячейки (формата полуслово-XH\$, слово-XF\$, X\$)	SINGREMENT, SDECREMENT	при инициализации модели 0 или задается картой INITIAL
Матрицы сохраняемых величин (MSAVEVALUES)	MH\$(a, b) MX\$(a, b)	Элемент строки "a", столбца "b". Полуслово. Элемент строки "a", столбца "b". Полное слово.	MSAVEVALUE INITIAL	
Списки пользователя (USERS CHAINS)	CAS CHS CMS CCS	Среднее число элементов в списке Текущее число элементов в списке Максимальное число элементов в списке Общее число входов	LINK, UNLINK	Значения СЧА собираются автоматически
Ключи (LOGICS)	LRS	Ключ сброшен (логическое 1) или ключ установлен (логическое 0)	LOGIC	Первоначальное состояние ключа при инициализации модели «сброшен» или «установлен», если использовать карту LINITIAL
Очереди (QUEUES)	QS QAS QMS QCS QZS QTS QXS	Текущая длина очереди Средняя длина очереди Максимальная длина очереди Общее число входов Количество нулевых входов Среднее время пребывания транзакта в очереди Среднее время пребывания транзакта в очереди без учета нулевых входов	QUEUE	Значения СЧА собираются автоматически
Памяти (STORAGES)	SS RS SRS SAS SMS SCS STS	Текущее содержимое памяти Число свободных единиц памяти Коэффициент использования Среднее содержимое памяти Максимальное содержимое памяти Общее число входов Среднее время занятости единицы памяти	ENTER, LEAVE	Значения СЧА собираются автоматически

Тип	СЧА	Определение	Блоки	Примечание
1	2	3	4	5
	SES SNES SFS SNFS	Память пуста (логические 0 или 1) Память не пуста (логические 0 или 1) Память заполнена (логические 0 или 1) Память не заполнена (логические 0 или 1)		
Приборы (FACILITES)	FS	Состояние устройства (логические 0 или 1)	SEIZE RELEASE PREEMPT RETURN	Состояние устройств устанавливается автоматически
	FIS	Прибор находится в состоянии прерывания (логические 0 или 1)		
	FNIS	Прибор не находится в состоянии прерывания (логические 0 или 1)		
	FNUS	Прибор не используется (логические 0 или 1)		
	FSS	Номер транзакта, занимающего прибор		
	FPS	Номер прерывающего транзакта		
	FRS	Коэффициент использования устройства		
	FCS	Общее число входов		
	FTS	Среднее время пребывания транзакта в устройстве		
Системные атрибуты (SYSTEM ATTRIBUTES)	AC\$1	Абсолютное время	RESET START TERMINATE	Относительное время после RESET обнуляется Содержимое счетчика уменьшается на число единиц, указанных в блоке TERMINATE
	C\$1	Относительное время		
	TG\$1	Содержимое счетчика завершения		
	RN\$j	Случайное число в интервале [0, 1], и используемое как аргумент функции, и случайное число в интервале [0, 999] в карте f[variable]		

Блоки GPSS

Название блока	Операнд						
	A	B	C	D	E	F	G
ADVANCE	к, СЧА *N время задержки	к, СЧА *N модификатор А					
ASSEMBLE	к, СЧА *N количество транзактов						
ASSIGN	к, СЧА *N [+,-] [номер парам.]	к, СЧА *N					
BUFFER	Нет операндов						
DECREMENT [вычесть]	к, СЧА *N Номер параметра	к, СЧА *N значение					
DEPART	к, СЧА *N Номер очереди	к, СЧА *N Вес [по умолчанию 1]					
ENTER	к, СЧА *N Номер памяти	к, СЧА *N Число Ячеек					
GATHER	к, СЧА *N Число транзактов						
CATE_ aux где aux: SE,SF,SNE,SNF-LR,LS-NI,NU,U,I-M,MN-	к, СЧА *N [номер: памяти, ключа, прибора, блока]	Метка, к, СЧА *N					
GENERATE	к, СЧА *N интервал времени	к, СЧА *N модификатор А	к, СЧА *N время входа 1- транзакта	к, СЧА *N число транзактов к,	СЧА *N приоритет	к, СЧА *N число параметров	к, СЧА *N тип параметра
INCREMENT [сложить]	к, СЧА *N номер параметра	к, СЧА *N значение					
INDEX	к, СЧА *N номер параметра	к, СЧА *N значение					
LEAVE	к, СЧА *N номер памяти	к, СЧА *N число ячеек (по умолчанию 1)					
LINK	к, СЧА *N	LIFO, P*N FIFO	метка, номер блока (альтернативный выход)				

Название блока	Операнд						
	A	B	C	D	E	F	G
LOCATE	к, СЧА*N номер транзакта	тип (X, XB, XF, XH, P, PH, PF, PB)	к, СЧА*N номер параметра или ячейки	метка, альтернативный выход			
LOGIC_aux где aux: S - установить I инвертировать R - сбросить	к, СЧА*N номер ключа						
LOOP	к, СЧА*N номер параметра	метка (выход при [p*N] = 0)					
MARK	к, СЧА*N номер параметра						
MATCH	к, СЧА*N (номер блока)						
PREEMPT	к, СЧА*N (номер прибора)						
PRINT	СЧА1	СЧА2	СЧА3	СЧА4	СЧА5	СЧА6	СЧА7
PRIORITY	к, СЧА*N приоритет	BUFFER					
QUEUE	к, СЧА*N [номер очереди]	к, СЧА*N [количество освобождающихся единиц, по умолчанию 1]					
RELEASE	к, СЧА*N [номер прибора]						
RETURN	к, СЧА*N [номер прибора]						
SAVEVALUE	к, СЧА*N [+,-] [номер прибора]	к, СЧА сохраняемое значение	тип ячейки [H, F, XH, XF по умолч. F]				

Название блока	Операнд						
	A	B	C	D	E	F	G
SDECREMENT [вычесть]	k, СЧА*N [номер ячейки]	k, СЧА*N значение	тип ячейки [H, F, XH, XF по умолчанию F]				
SEIZE	k, СЧА*N [номер прибора]						
SINCREMENT [сложить]	k, СЧА*N [номер ячейки]	k, СЧА*N значение	тип ячейки [H, F, XH, XF по умолч. F]				
SPLIT	k, СЧА*N [число копий]	Метка [блок вывода]	k, СЧА*N [номер параметра]	k, СЧА*N [число параметров]			
TERMINATE	k число транзактов (по умолчанию 0)						
TEST_aux где aux: E, NE, LE, G, GE	k, СЧА*N	k, СЧА*N Метка альтернативного блока					
TRACE	Без операндов						
TRANSFER	FN P BOTH, СЧА*N, k SBR	Номер функции Номер параметра Метка блока Номер параметра, метка блока	Метка блока Метка блока				

Название блока	Операнд						
	A	B	C	D	E	F	G
UNLINK	ALL ,_	Номер или метка блока					
	PICK	Номер или метка блока	Номер или метка блока	Номер или метка блока	Значение шага k		
	k, СЧА*N номер цепи пользователя	метка блока	метка блока	Номер или метка блока			
USING	k, СЧА*N номер транзакта	k, СЧА*N номер параметра	k, СЧА*N число транзактов	k, СЧА*N число параметров	k, СЧА*N номер параметра	k, СЧА*N значение	Альтернативный выход
UNTRACE	Нет операндов						

Карты GPSS

Название карты	Операнд						
	A	B	C	D	E	F	G
END	нет операндов						
EQU	k	(-32766<k<32766)					
DQTABLE	k номер очереди	k верхняя граница 1 интервала	k величина интервала	k число интервалов			
DTABLE	k, СЧА*N аргумент	k верхняя граница 1 интервала	k величина интервала	k число интервалов			
FUNCTION	СЧА*N аргумент	тип функции [C,D]					
FVARIABLE	Логические операции: см. VARIABLE , арифметические операции: + сложить, - вычесть, / разделить, (), * умножить, @ сложить по модулю						
INITIAL	Номер ячейки [XH,XF,X]	k,...					
JOBTAPE	Имя файла	метка					
LINITIAL	LSSN1	LSSN2	LSSN3	NSSN4	LSSN5	LSSN6	LSSN7
REALLOCATE	Мнемоническое обозначение	k, ...		(повторное описание)			
RESET	нет операндов						
REWIND	k количество файлов						
RMULT	RNS1	RNS2	RNS3	RNS4	RNS5	RNS6	RNS7
TABLE	СЧА*N [-], IA, RT аргумент	k верхняя граница 1 интервала	величина интервала k	число интервалов k			
VARIABLE	Логические операции: = (равно), < (меньше), > (больше), <= (меньше или равно), >= (больше или равно), & (и), \ (или); арифметические операции: см. FVARIABLE						
QTABLE	Номер очереди [-]	k верхняя граница 1 интервала	размер интервала k	число интервалов k			
START	k счетчик транзактов	отмена печати NP	k интервал выдачи печати	1-выдача сообщений о тразактах			
STORAGE	S*N	k значение	, разделитель				
SIMULATE	нет операндов						
WDQTABLE	номер очереди k	k верхняя граница 1 интервала	размер интервала k	число интервалов k			
WDTABLE	номер очереди k	k верхняя граница 1 интервала	размер интервала k	число интервалов k			

ПРИМЕЧАНИЕ: N может принимать значения:

1. *num, где num - номер параметра транзакта
2. СЧА\$к, где k - константа