С.И. Матюшенко, С.С. Спесивов

ОСНОВЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ В СРЕДЕ GPSS WORLD

Москва 2006

Утверждено
Редакционно-издательским советом
Российского университета дружбы
народов

Рецензенты:

д.ф.-м. н., профессор Шоргин С. Я.
д.ф.-м. н., профессор Саксонов Е.А.

Матюшенко С.И., Спесивов С.С. **Основы имитационного моделирования в среде GPSS World**: Учебное пособие.- М.: Изд-во РУДН, 2006. - 112 с.

В предлагаемом учебном пособии излагается принципы имитационного моделирования и содержатся краткие сведения об основных элементах языка GPSS World. Рассматриваемый материал иллюстрируется примерами моделирования различных систем, содержащими пояснения по применению соответствующих элементов языка.

Пособие рассчитано на студентов, обучающихся по специальности «Прикладная математика и информатика», а также студентов и аспирантов других специальностей, использующих в своей работе методы имитационного моделирования.

Введение

Имитационное моделирование - мощный универсальный метод исследования систем, функционирование которых зависит от воздействия случайных факторов. Применение современных вычислительных средств в совокупности с универсальными языками программирования позволяют исследователю достичь значительных результатов при изучении сложных объектов, избегая многих ограничений и допущений, которые неизбежны при аналитическом моделировании.

Весьма эффективным и достаточно простым языком имитационного моделирования является GPSS (General Purpose Simulating System – общецелевая система моделирования). Его развитие началось в конце 50-х годов прошлого века. В конце 80-х годов одну из версий языка, а именно GPSS/360, авторы этого пособия достаточно успешно использовали при моделировании информационно-вычислительных систем и сетей передачи данных. Приобретенный опыт явился основой для написания учебно-методического пособия [1], в котором были изложены основные принципы моделирования, а также отдельные элементы языка GPSS/360 и рассмотрен ряд моделей. В настоящем учебном пособии рассматривается современная версия GPSS, разработанная компанией Minuteman (США) и получившая название GPSS World [2]. По сравнению с предыдущими версиями, эта версия дополнена новыми объектами языка GPSS. Кроме этого она позволяет включать в себя элементы, написанные на языке Plus. Данное обстоятельство делает GPSS World более универсальной системой и дает возможность взаимодействовать с другими программными средствами.

Настоящее учебное пособие соответствует программе курса «Стохастическое моделирование», который читается

авторами для студентов специальности «Прикладная математика и информатика». В первой главе излагаются основные принципы статистического моделирования. Описанию элементов языка GPSS World посвящена вторая глава. В этой главе также содержится большое количество примеров моделирования различных систем. Распечатки моделей и результатов моделирования с подробными комментариями и пояснениями, позволяющими понять логику моделирования, приводятся в Приложении.

Авторы надеются, что данное учебное пособие будет полезным для тех, кто использует имитационное моделирование в своих научных исследованиях, и особенно для тех, кто делает первые шаги в этой области.

ГЛАВА 1. ОСНОВЫ СТАТИСТИЧЕСКОГО МОДЕЛИРОВАНИЯ

1.1. Метод статистического моделирования

Предположим, что вам необходимо построить модель некоторой системы, функционирование которой зависит от ряда случайных факторов. Если объект моделирования имеет сложную структуру, либо сложный алгоритм функционирования, то обойтись только аналитическими методами весьма затруднительно, а иногда и вовсе невозможно. В этом случае полезно использовать метод статистического моделирования, который также называют методом Монте-Карло.

Суть метода состоит в том, что вместо описания случайных явлений аналитическими зависимостями проводится розыгрыш этих случайностей. Для проведения розыгрыша используют генераторы случайных чисел, которые строятся аппаратными и программными методами. Осуществляя многократную имитацию, накапливают материал, который затем обрабатывают методами математической статистики.

Чтобы понять суть метода, рассмотрим следующий пример [3].

Пример 1.1. На некоторую цель бомбардировщики сбрасывают n бомб. Каждая бомба поражает область в виде круга радиусом r (рис. 1.1). Цель считается пораженной, если одновременно бомбами накрыто K процентов площади S. Найти вероятность поражения цели.

Аналитически решить эту задачу очень трудно. Покажем, как ее можно решить методом статистического моделирования.

Наложим координатную сетку на всю возможную область попадания бомб.

С помощью генератора случайных чисел разыграем n пар чисел, являющихся координатами попадания бомб. Опишем возле каждой точки круг радиусом r (рис. 1.2) и определим заштрихованную площадь поражения. Если заштрихованная площадь будет составлять не менее K процентов от всей площади цели S, то цель считается

пораженной, а испытание успешным. В противном случае цель не будет поражена, а испытание является неуспешным. Многократное

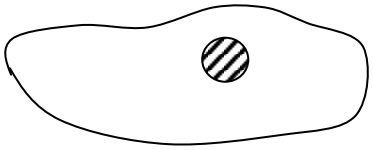

Рис. 1.1

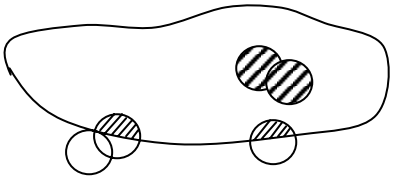

Рис. 1.2

повторение имитации позволит оценить искомую вероятность. При этом заметим, что вопрос о количестве повторений и соответственно о точности результата, мы пока не рассматриваем.

Итак, метод статистического моделирования – это такой математический метод, при котором сама случайность непосредственно включена в процесс моделирования и является его важным элементом. Каждый раз, когда в ход выполнения некоторой операции вмешивается случайный фактор, его влияние моделируется с помощью розыгрыша.

1.2. Моделирование случайных событий и дискретных случайных величин

Моделирование отдельного случайного события. Пусть необходимо смоделировать появление некоторого события *A,* вероятность наступления которого равняется *P(A).* Для этого используем генератор, который разыгрывает случайные числа, равномерно распределенные на интервале (О, 1). Обозначим через R – результат розыгрыша. Событие A будем считать свершившимся, если $R < P(A)$. В противном случае будем считать, что произошло $\overline{A}$ (рис.1.3).

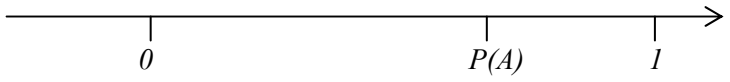

Рис. 1.3

Моделирование группы несовместных событий. Пусть $A_1, A_2, \ldots, A_n$ – несовместные события, наступающие с вероятностями $P(A_1), P(A_2),\ldots,P(A_n)$ и составляющие полную группу событий, т.е. $\sum_{i=1}^{n} P(A_i) = 1$

Разобьем интервал (0,1) так, как показано на рисунке 1.4 и с помощью генератора случайных чисел, распределенных равномерно на (0,1), получим число R .

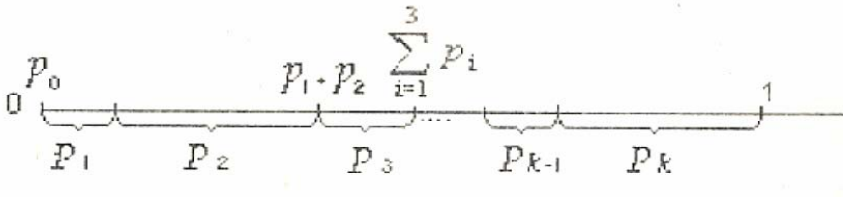

Рис.1.4.

Попадание числа R в интервал от $\sum_{i=o}^{k-1} P(A_i)$ до $\sum_{i=o}^{k} P(A_i)$,

где $P(A_0)$ полагаем равным нулю, будет означать , что произошло событие A_k.

Моделирование дискретной случайной величины. Рассмотрим дискретную случайную величину X с рядом распределения:

x_1	x_2	. . .	x_n
p_1	p_2	. . .	p_n

Случайной величине X поставим в соответствие полную группу событий:

$A_1=\{X=x_1\}$, . . . , $A_n=\{X=x_n\}$ с вероятностями p_1 , . ..,p_n . Данное соответствие позволяет моделировать случайную величину *X* также, как и полную группу несовместных событий.

Моделирование условного события. Рассмотрим два события A и B, происходящие с вероятностями *P(A)* и *P(B)* соответственно. Для того, чтобы смоделировать условное событие *A/B,* сначала моделируем событие *B* с вероятностью *P(B)* так как мы это делали при моделирования отдельного события. Если событие *B* произошло, то моделируем наступление события *A*. Если событие *B* не произошло, то очевидно и наступление события A моделировать нет смысла, поскольку оно невозможно.

1.3. Моделирование непрерывных случайных величин

При моделировании непрерывных случайных величин используется метод обратных функций. Суть этого метода поясним на конкретных примерах.

Моделирование случайной величины, имеющей равномерное на *(a,b)* распределение

Известно, что функция распределения (ФР) F(x) в этом случае имеет вид:

$$F(x)=\frac{x-a}{b-a}, \quad x\in (a,b),$$

Обозначим через *R* значение *F(x),* которое можно разыграть с помощью генератора случайных чисел. Следовательно,

$$R=\frac{x-a}{b-a}$$

Выражая *x,* получим:

$$x(R) = (b - a)R + a$$

Очевидно, что данное выражение позволяет с помощью генератора случайных чисел из интервала *(0,1)* получать конкретные реализации случайной величины (с.в.), имеющей равномерное на *(a,b)* распределение.

Моделирование случайной величины, распределенной по экспоненциальному закону

Рассмотрим с.в. *X*, имеющую экспоненциальное распределение с параметром λ, и обозначим через *F(x)* ее ФР, т.е.:

$$F(x) = 1 - e^{-\lambda x},\ x > 0$$

Поскольку *F(x)* представляет собой вероятность, то мы можем разыграть ее значение с помощью генератора случайных чисел на *(0,1)*. Результат розыгрыша обозначим через *R*. Таким образом,

$$R = 1 - e^{-\lambda x}$$

Выражая *x*, получим:

$$x(R) = -\frac{1}{\lambda}\ell n(1 - R)$$

Далее рассмотрим случай, когда $\lambda = 1$, и выполним линейную аппроксимацию функции *x(R)* по 24-ем точкам:

R	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,75	0,8	0,84	0,88
x(R)	0	0,1	0,222	0,355	0,509	0,69	0,915	1,2	1,38	1,6	1,83	2,12

R	0,9	0,92	0,94	0,95	0,96	0,97	0,98	0,99	0,995	0,998	0,999	0,9998
x(R)	2,3	2,52	2,81	2,99	3,2	3,5	3,9	4,6	5,3	6,2	7	8

Мы не будем подробно останавливаться на вопросе о том, почему именно эти точки и именно в таком количестве избраны в качестве узлов аппроксимации. Желающие получить ответ на этот вопрос могут воспользоваться книгой [4]. Мы лишь отметим, что данный выбор в дальнейшем будет отвечать нашим требованиям к точности результатов моделирования.

Итак, разыгрывая значения ФР *F(x)* с помощью генератора случайных чисел, мы тем самым попадаем в интервал на оси абсцисс, которому соответствует конкретная реализация рассматриваемой с.в. (рис. 1.5).

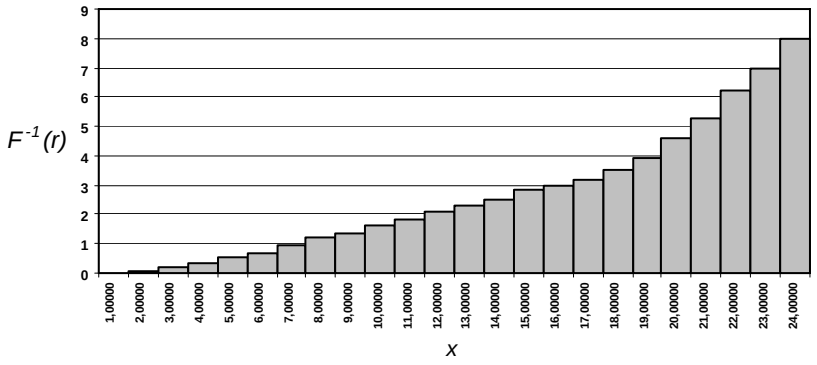

Рис.1.5.

Теперь остается лишь заметить, что когда $\lambda \neq 1$, полученное путем розыгрыша значение с.в., следует домножить на $1/\lambda$..

Моделирование нормально распределенной случайной величины

Обозначим через X с.в., распределенную по стандартному нормальному закону, а через Y – с.в., имеющую нормальное распределение с параметрами a и σ^2.

Поскольку между X и Y существует очевидная связь, которая выражается в виде соотношения

$$Y = \sigma^2 X + a \qquad (1.1)$$

нам достаточно рассмотреть задачу моделирования с.в. X.

Решить данную проблему можно разными способами, но мы ограничимся рассмотрением двух наиболее часто встречающихся.

Итак, рассмотрим 12 независимых с.в. $R_1, \ldots R_{12}$, имеющих равномерное на (0,1) распределение.

Теперь воспользуемся центральной предельной теоремой, согласно которой ФР с.в.

$$Z = \frac{\frac{1}{12}\sum_{i=1}^{12} R_i - MR_i}{\sqrt{\frac{DR_i}{12}}}$$

приближается к ФР стандартного нормального закона при всех значениях аргумента.

Учитывая, что $MR_i = \frac{1}{2}$, $DR_i = \frac{1}{12}$, $i=1, \ldots ,12$, получим:

$$Z = \sum_{i=1}^{12} R_i - 6 \qquad (1.2)$$

Очевидно, что $MZ=0, DZ=1$. Таким образом, разыгрывая с помощью генератора случайных чисел значения с.в. $R_1, \ldots R_{12}$ и подставляя эти значения в формулу (1.2), получим реализацию с.в., имеющей стандартное нормальное распределение. Заметим, что вопрос о точности предложенного метода мы здесь не

рассматриваем. Любознательному читателю по данному вопросу рекомендуем обратиться к работе [5].

Рассмотрим еще один способ генерации с.в. X. Именно этот метод и будет использоваться нами при построении модели на GPSS. Используя 25 точек, приведенных в следующей таблице:

x	-5	-4	-3	-2,5	-2	-1,5	-1,2	-1	-0,8
F(x)	0	0,00003	0,00135	0,00621	0,02275	0,06681	0,11507	0,15866	0,21186

x	-0,6	-0,4	-0,2	0	0,2	0,4	0,6	0,8	1
F(x)	0,27425	0,34458	0,42074	0,5	0,57964	0,65542	0,72575	0,78814	0,84134

x	1,2	1,5	2	2,5	3	4	5
F(x)	0,88493	0,93319	0,97725	0,99865	0,99865	0,99997	1

построим линейную аппроксимацию ФР *F(x)* стандартного нормального закона (рис. 1.6). Далее перейдем к обратной функции $F^{-1}(R)$, график которой представлен на рисунке 1.7. Теперь для того, чтобы получить конкретную реализацию с.в. X, нам достаточно разыграть значение аргумента R, являющееся по сути вероятностью, т.е. числом из интервала *(0,1)*.

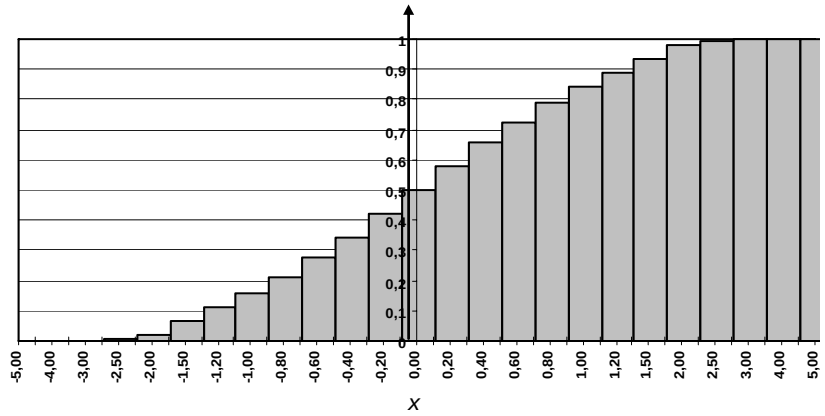
1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
-5,00
-4,00
-3,00
-2,50
-2,00
-1,50
-1,20
-1,00
-0,80
-0,60
-0,40
-0,20
0,00
0,20
0,40
0,60
0,80
1,00
1,20
1,50
2,00
2,50
3,00
4,00
5,00
x

Рис.1.6.

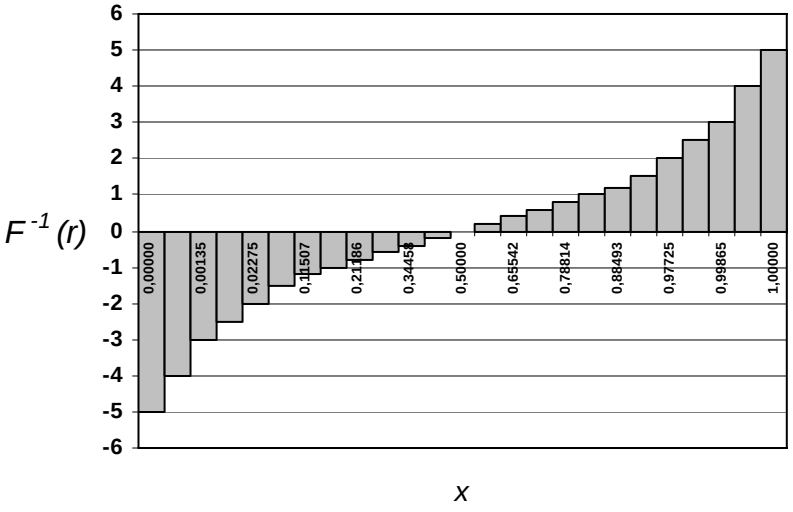
6
5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
F -1 (r)
0,00000
0,00135
0,02275
0,15507
0,21186
0,34458
0,50000
0,65542
0,78814
0,88493
0,97725
0,99865
1,00000
x

Рис.1.7.

В заключение заметим, что для того, чтобы получить значение с.в., имеющей распределение с произвольными параметрами a и σ^2, достаточно воспользоваться формулой (1.1).

ГЛАВА 2. ОСНОВНЫЕ ЭЛЕМЕНТЫ GPSS

2.1. Общая структура модели на языке GPSS

Язык GPSS состоит из трех основных типов элементов: транзактов, являющихся динамическими элементами, блоков, являющихся статическими элементами, и операторов, служащих для задания исходных данных и управления процессом моделирования.

Блоки языка GPSS представляют собой подпрограммы, написанные на макроассемблере или на языке Си, и содержат набор параметров (операндов) для обращения к ним. Как и во всех языках моделирования в GPSS существует внутренний механизм передачи управления, который реализуется в модельном времени, что дает возможность отобразить динамические процессы в реальных системах. Передача управления от блока к блоку в GPSS-программах реализуется с помощью движения транзактов в модельном времени. Обращение к подпрограммам блоков происходит через движение транзактов.

Содержательное значение транзактов определяет разработчик модели. Именно он устанавливает аналогию между транзактами и реальными динамическими элементами моделируемой системы. Такая аналогия никогда не указывается транслятору GPSS, она остается в воображении разработчика моделей. В табл. 2.1 приведены примеры аналогий между транзактами и элементами реальных систем.

Таблица 2.1

Система	Элементы систем, которые моделируются транзактами
Магазин	Покупатель
Автомобильное шоссе	Автомобиль
Склад	Заявка

С точки зрения программы - транзакт это структура данных, которая содержит следующие поля: имя или номер транзакта; время появления транзакта; текущее модельное время; номер блока, в котором находится транзакт; номер блока, куда он продвигается; момент времени начала продвижения; приоритет транзакта; параметры транзакта: PI, P2,... и т.д.

В языке GPSS все транзакты нумеруются по мере их появления в модели. Параметры транзактов отображают свойства моделируемого динамического объекта. Например, если моделируется движение автомобилей на участке дороги, то параметрами транзакта (автомобиля) в зависимости от целей моделирования могут быть скорость, тормозной путь, габариты и др.

Каждый транзакт занимает некоторые объем памяти ЭВМ. После того, как он закончит свое движение по блокам модели, его необходимо уничтожить для освобождения памяти, чтобы избежать ее переполнения. Поскольку транслятору не известно, сколько транзактов одновременно будет находиться в модели, то память под транзакты выделяется динамически.

Таким образом, при начале моделирования в GPSS-модели не существует ни одного транзакта. В процессе моделирования транзакты входят в модель в определенные моменты времени, соответствующие логике функционирования моделируемой системы. Таким же образом транзакты покидают модель в зависимости от специфики моделирования. В общем случае в модели существует несколько транзактов, но в каждый момент времени движется только один из них.

Если транзакт начал свое движение, он передвигается от блока к блоку по пути, указанному блок-схемой (логикой работы модели). В тот момент, когда транзакт входит в блок, вызывается соответствующая этому блоку подпрограмма. Далее транзакт (в общем случае) пытается войти в следующий блок. Его перемещение продолжается до тех пор, пока не выполнится одно из следующих возможных условий:

- транзакт входит в блок, функцией которого является задержка транзакта на некоторое время;

- транзакт входит в блок, функцией которого является удаление транзактов из модели;

- транзакт пытается войти в блок, который по некоторым причинам не может его принять.

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

2.2. Таймер модельного времени

Таймер модельного времени (ТМВ) - это "внутренние" часы интерпретатора, с помощью которых отслеживается последовательность событий, происходящих в модели. В момент начала моделирования таймер устанавливается в нуль. После очередного приращения таймера интерпретатор в определенной последовательности продвигает по блокам модели находящиеся в ней транзакты (порядок продвижения транзактов будет рассмотрен позднее). Если ни один транзакт более не может быть продвинут, то происходит следующее приращение таймера до момента времени, когда возникает событие, разрешающее последующее продвижение транзактов.

Отметим некоторые важные особенности GPSS и его таймера. Таймер GPSS регистрирует только целые значения, т.е. события в системе могут происходить только в целочисленные моменты времени (за исключением языка GPSS World, где время может иметь действительные значения). Единицу времени, регистрируемую таймером, определяет разработчик модели. Она задается в неявном виде в форме исходных данных, вводимых в модель.

2.3. Общая структура блоков и операторов

Язык GPSS содержит более 40 блоков и около 20 операторов, каждый из которых имеет свое уникальное имя.

Формат GPSS-блоков такой:

[Номер строки] [< Метка >] < Операция > < Операнды >
<; Комментарии >.

При описании форматов квадратные скобки [ ] указывают на необязательность поля.

Номер строки. Обязательное поле для GPSS/PC (в GPSS World - игнорируется). Начинается с первой позиции строки. Представляет собой десятичное число.

Метка (имя блока). Содержимым поля является имя. Имя представляет собой алфавитно-цифровую последовательность длиной до 20 символов в GPSS/PC и до 250 символов в GPSS World, которая начинается с буквы. Допускается использование символов только латинского алфавита, цифр и знака подчеркивания. В некоторых операторах это поле является обязательным.

Операция. Операциями блоков являются глаголы, которые описывают основные функциональные назначения блоков. Каждый из блоков характеризируется своим собственным предписанным ему глаголом.

Операнды. Блоки могут иметь операнды. Операнды блоков задают информацию, специфичную для действия данного блока. Число операндов блока зависит от типа блока. В блоках не может использоваться больше семи операндов. Операнды в общем случае обозначаются символами: A, B, C, D, E, F, G. Значения операндов определяются типом блока. Одни операнды некоторых блоков должны быть определены всегда, а другие могут задаваться или не задаваться (т.е. являются необязательными). Операнды следуют один за другим и отделяются запятыми или одним пробелом. Если операнд пропущен, то вместо него ставится запятая. Между операндами не должно быть более одного пробела, так как это будет означать, что операнды закончились и интерпретатор прекращает чтение строки.

Комментарии. Необязательное поле. Комментарии отделяются от поля операндов символом «;». Допускается запись комментария с начала строки. В этом случае в первой позиции строки ставится символ «;» или «*». В GPSS/PC допускаются комментарии с использованием заглавных или строчных букв только латинского алфавита, в GPSS World также допускается использование символов кириллицы.

Строка описания блока может содержать до 79 символов в GPSS/PC и до 250 символов в GPSS World.

Именами и метками не могут быть названия или начальные символы названий блоков, операторов, команд и стандартных числовых атрибутов. Во избежание конфликтов с ключевыми словами рекомендуется в именах использовать символ подчеркивания.

2.4. Блок GENERATE

GENERATE – это блок, через который транзакты входят в модель. Число этих блоков в модели может быть произвольным. Интервал времени между последовательными появлениями транзактов из блока GENERATE называется интервалом поступления. Когда транзакт входит в модель через блок GENERATE, интерпретатор планирует время поступления следующего транзакта путем розыгрыша случайной величины в соответствии с требуемым распределением интервалов поступления и последующим добавлением полученного значения к текущему значению ТМВ. При достижении этого значения времени следующий транзакт поступает в модель и т.д.

В начале моделирования все транзакты нумеруются, начиная с единицы и располагаются в так называемом пассивном буфере. Число транзактов конечно, но достаточно для решения большинства задач. В принципе число транзактов в пассивном буфере при необходимости может быть увеличено или уменьшено, но этот вопрос рассматриваться не будет,

Когда нужно ввести новый транзакт в модель, интерпретатор достает первый по порядку транзакт из пассивного буфера и он через блок GENERATE, поступает в модель.

Значения операндов блока GENERATE приведены в табл. 2.2. Операнд A задает среднюю длину интервала поступления. Если в качестве операнда B указывается число, то длина интервала поступления распределена равномерно на интервале [A-B, A+B].

Таблица 2.2.

операнд	значение	значение по умолчанию
A	Среднее время	0
B	Модификатор разброса или модификатор функции	0
C	Интервал смещения	нет
D	Ограничитель	бесконечность
E	Уровень приоритета	0

Случай, когда в качестве операнда B указывается модификатор функции, будет рассмотрен после введения функций. Операнд C определяет время поступления первого транзакта в модель. Операнд D задает предельное количество транзактов, которые могут войти в модель через данный блок GENERATE. Операнд E определяет уровень приоритета транзакта, который может изменяться от 0 до 127.

Каждый транзакт может иметь до 100 параметров. Разработчик может задавать, изменять их и использовать в соответствии с логикой моделирования. Более подробно этот вопрос будет рассмотрен позже.

Ниже приведены примеры использования блока GENERATE.

Пример 2.1. Задание равномерного закона распределения:

```
GENERATE    6,4
```

Операнды: A = 6, B = 4. Интервал времени поступления является случайным числом со средним значением 6 и полем допуска 8, то есть он может приобретать только одно из девяти разных значений: 2,3,4,5,6,7,8,9,10.

Пример 2.2. Задание детерминированного значения интервалов поступления.

```
GENERATE    10
```

Операнды: A = 10, B = 0 (по умолчанию). Транзакты входят в модель каждые 10 единиц модельного времени.

Пример 2.3. Генерирование одного транзакта.

GENERATE ,,,1

Операнды: A = B = C = 0 (по умолчанию), D = 1. В нулевой момент в модель входит один транзакт.

2.5. Блок TERMINATE, команда START.

Транзакты удаляются из модели, попадая в блок TERMINATE. В этот блок может войти любой транзакт, который пытается это сделать. В модели может быть любое число блоков TERMINATE. Проходя через блок TERMINATE, транзакты попадают в пассивный буфер для последующего их использования, причем первоначальная их нумерация сохраняется (см. описание блока GENERATE).

Блок TERMINATE имеет только один операнд (табл.2.3), являющийся указателем уменьшения счетчика завершений.

Таблица 2.3

Операнд	Значение	Значение по умолчанию
A	Уменьшение счетчика числа завершений	0

Счетчик числа завершений - это область памяти интерпретатора, в которой хранится целое положительное число, записанное в начале моделирования. В процессе моделирования транзакты попадают в блоки TERMINATE и уменьшают значение счетчика числа завершений на величину, равную значению операнда A . Когда его значение становится равным нулю или отрицательным, моделирование прекращается. Следует обратить внимание на то, что в модели может быть много блоков TERMINATE, но счетчик числа завершений только один.

Начальное значение счетчика числа завершений задается с помощью операнда A команды START. Формат этой команды выглядит следующим образом:

START A, [B], [C], [D]

Опреранды B, C и D для нас не существенны, поэтому рассматриваться здесь не будут. Команда START используется для инициации начала моделирования. Обращение к команде START в GPSS World осуществляется в диалоговом режиме из раздела Command.

2.6. Блоки SEIZE и RELEASE.

В языке GPSS существуют элементы, называемые приборами. Понятие "прибор" в GPSS совпадает с соответствующим понятием в теории массового обслуживания. Элементами, требующими обслуживания на приборах, являются транзакты. Приборы занимаются и освобождаются транзактами, проходящими через блоки SEIZE и RELEASE соответственно. Эти блоки имеют единственный операнд A (табл.2.4), являющийся именем занимаемого или освобождаемого прибора.

Таблица 2.4

Операнд	Значение	Значение по умолчанию
A	Имя (символическое или числовое) прибора	Ошибка

Имя прибора может быть либо символическим (правила составление таких имен аналогичны правилам составления символических имен блоков), либо числовым, т.е. порядковым номером прибора в модели. Рекомендуется в модели придерживаться одного способа задания имен.

Когда транзакт пытается войти в блок SEIZE, осуществляется проверка, свободен ли прибор A. Если прибор занят, то вход в

блок SEIZE запрещается и транзакт ожидает освобождения прибора, занимая блок, из которого он пытался войти в блок SEIZE.

Блок RELEASE никогда не запрещает вход транзакта. Но было бы нелогично освободить свободный прибор или прибор, занятый другим транзактом. В обоих этих случаях выдается сообщение об ошибке и моделирование прекращается.

Отметим, что использование блока SEIZE устанавливает и сам факт существования соответствующего прибора, т.е. нет необходимости заранее, до применения блока SEIZE, определять прибор, используемый в этом блоке.

При моделировании по всем приборам модели интерпретатором GPSS собирается статистика, которая распечатывается в конце моделирования. Необходимые пояснения даны в приводимых ниже примерах.

2.7. Блок ADVANCE

Для реализации задержки продвижения транзакта в течение некоторого интервала времени (в том числе и для реализации обслуживания на приборе) в GPSS существует блок ADVANCE. Информация, необходимая для описания длительности задержки, задается операндами A и B этого блока (табл.2.5).

Таблица 2.5

Операнд	Значение	Значение по умолчанию
A	Среднее время	0
B	Модификатор разброса или модификатор функции	0

Аналогично блоку GENERATE операнд A задает среднее время задержки. Если указано числовое значение операнда B, то время задержки распределено равномерно на интервале [A-B, A+B]. Случай, когда в качестве операнда B указывается модификатор функции, будет рассмотрен после введения функций.

Блок ADVANCE никогда не препятствует входу транзакта. Любое их число может находиться в этом блоке одновременно, не влияя друг на друга.

2.8. Блоки QUEUE и DEPART.

В большинстве систем массового обслуживания (СМО) число приборов ограничено, в силу чего перед ними обычно расположены накопители, где размещаются заявки, ожидающие своей очереди на обслуживание. Нахождение характеристик таких очередей является важной задачей при исследовании СМО. GPSS обеспечивает возможность с помощью средства, называемого регистратором очереди, собирать статистику очередей, имеющихся в моделируемой системе.

Разработчик вносит регистратор очереди в модель с помощью пары взаимодополняющих блоков QUEUE и DEPART. Операнд A этих блоков (табл.2.6) задает имя соответствующей очереди.

Таблица 2.6

Операнд	Значение	Значение по умолчанию
A	Имя (символическое или числовое) очереди	Ошибка

Имя может быть либо символическим, либо числовым, но, как и ранее, рекомендуется придерживаться одного способа задания имен. Рассматриваемые блоки могут иметь также операнд B, но он используется чрезвычайно редко и поэтому не рассматривается.

При входе транзакта в блок QUEUE выполняются следующие действия:

- счетчик входов для данной очереди (счетчик числа транзактов, вошедших в данный блок QUEUE) увеличивается на единицу;
- счетчик текущего содержимого очереди также увеличивается на единицу;
- осуществляется «привязка» транзакта к очереди, т.е. запоминается тот факт, что транзакт вошел в данный блок QUEUE;

- запоминается модельное время присоединения транзакта к очереди, т.е. осуществляется "привязка" по времени.

Транзакт перестает быть элементом данной очереди, когда он переходит в блок DEPART с операндом A, являющимся именем этой очереди. Когда это происходит, интерпретатором выполняются следующие действия:

- счетчик текущего содержимого соответствующей очереди уменьшается на единицу;
- используя «привязку» по времени, интерпретатор определяет время, проведенное транзактом в очереди. Если это время равно нулю, то такой транзакт называется транзактом с нулевым пребыванием в очереди и соответствующим образом изменяется счетчик «нулевых вхождений»;
- ликвидируется "привязка" транзакта к очереди.

В конце моделирования интерпретатор автоматически распечатывает собранную по очередям статистику, пояснения по которой даны в приводимых примерах.

2.9. Пример моделирования 1

2.9.1. Постановка задачи. Рассмотрим модель однолинейной СМО с накопителем бесконечной емкости. На систему поступает рекуррентный поток заявок, промежутки между приходами которых распределены равномерно на интервале 18+6. Длительности обслуживания заявок являются независимыми в совокупности случайными величинами, равномерно распределенными на интервале 16±6. Моделирование необходимо закончить, когда через систему пройдет 1000 заявок.

2.9.2. Логика моделирования. Распечатка модели и результатов приведена в приложении 2.1.1. Модель системы состоит из семи блоков. Каждый блок снабжен кратким комментарием для лучшего понимания модели.

Рассмотрим подробнее программу моделирования. Блок GENERATE моделирует поступление заявок в систему, а сами заявки моделируются транзактами. В блоке явно заданы первые два операнда, говорящие о том, что промежутки времени между

приходами заявок распределены равномерно на интервале 18±6. Поступление заявки в накопитель моделируется поступлением транзакта в блок QUEUE. Единственный операнд этого блока QUE задает имя очереди в символьном виде. Если накопитель не пуст и/или прибор занят, то заявка остается в накопителе, ожидая своей очереди на обслуживание (запрещается вход транзакта в блок SEIZE и он остается в блоке QUEUE). При освобождении прибора заявка занимает его, покидая при этом накопитель. Этот процесс моделируется последовательностью блоков SEIZE - DEPART. Операнд A блока SEIZE задает символическое имя прибора, а блока DEPART - символическое имя той же очереди, что операнд A блока QUEUE. Длительность обслуживания на приборе моделируется блоком ADVANCE . Явное задание операндов A и B в этом блоке указывает на то, что время обслуживания имеет равномерное распределение на интервале 16±6. Процесс освобождения прибора моделируется прохождением транзакта через блок RELEASE, операнд A которого указывает символическое имя освобождаемого прибора. Напомним, что прохождение транзакта через блок RELEASE позволяет следующему по порядку транзакту (если такой имеется) войти в блок SEIZE, т.е. занять прибор. Таким образом по умолчанию подразумевается дисциплина FIFO выбора транзактов из очереди.

Освободившая прибор заявка покидает систему, что моделируется прохождением транзакта через блок TERMINATE. При этом из счетчика числа завершений вычитается единица (значение операнда A блока TERMINATE). Начальное значение этого счетчика задается в интерактивном режиме командой START и по условию задачи полагается равным 1000. После выхода из системы 1000 заявок счетчик числа завершений обнулится и моделирование закончится.

2.9.3. Результаты моделирования. По окончании прогона модели интерпретатор распечатывает стандартную статистику. Вначале указывается значение TMB (18063.296), при котором завершился процесс моделирования.

Вслед за строкой времени дается информация о блоках: номер блока (LOC), название блока (BLOCK TYPE), счетчик числа входов в блок (ENTRY COUNT), указывающего на количество транзактов, вошедших в данный блок за все время моделирования, счетчик текущего содержимого блоков (CURRENT COUNT), указывающий на количество транзактов, находящихся в момент окончания моделирования в данном блоке. Эта информация может быть полезна при проверке логики моделирования.

Далее следует раздел FACILITIES, в котором собраны статистические данные по всем приборам, встречающимся в модели. Первый столбец раздела указывает на имена приборов. В рассматриваемом примере использовался один прибор SERV. В втором столбце дается общее число транзактов, занимавших прибор - 1001, что соответствует числу входов в блок SEIZE (блок №3). В третьем столбце приводится значение загрузки прибора (UTIL.), равное 0.882.

В следующем столбце дается значение среднего времени пребывания транзактов на приборе (AVE.TIME), которое в нашем примере равно 15.920. Столбец с названием AVAIL, содержит информацию о занятости прибора в момент остановки моделирования (0 - прибор свободен, 1 - прибор занят), а в столбце (OWNER) выдается номер транзакта, обслуживанием которого занят прибор в этот момент (если таковой имеется). В нашем примере это транзакт с номером 1001. Информация, содержащаяся в остальных столбцах для нас несущественна.

В разделе QUEUE приводится статистика по всем регистраторам очередей, внесенным в модель разработчиком. В рассматриваемом примере статистика собиралась по одной очереди с именем QUE (первый столбец). Во втором столбце выдается максимальное число транзактов в очереди в течение всего времени моделирования. В примере это значение равно 3. Далее, в столбце CONT. дается текущее содержимое очереди. В нашем примере оно равно 1, т.к. один транзакт в момент

остановки моделирования находится в блоке SEIZE и освободит очередь только после прохождения блока DEPART. В столбце ENTRY приводится значение числа транзактов, вошедших в очередь. Оно совпадает с счетчиком числа входов в блок QUEUE, равным 1001. В столбце ENTRY(O) регистрируется число транзактов с нулевым временем пребывания в очереди (482), а следующий столбец AVE.CONT выдает среднее число транзактов в очереди (0.172) в течении всего времени моделирования. Следующие два столбца AVE.TIME и AVE.(-O) дают среднее время пребывания транзакта в очереди соответственно с учетом транзактов с нулевым временем пребывания (3.095) и без их учета (5.969).

Заметим, что интерпретация полученных статистических данных в терминах моделируемой СМО лежит на разработчике, но обычно не вызывает каких-либо трудностей.

2.10. Цепи текущих и будущих событий. Логика работы интерпретатора

Как уже было сказано ранее, работу интерпретатора можно условно представить следующим образом: в соответствии с некоторым правилом выбирается транзакт и продвигается по модели от блока к блоку, пока это возможно, затем выбирается некоторый другой транзакт, также продвигается по модели и т.д.

Движение транзакта определяется расположением блоков в модели и местоположением самого транзакта в модели. Правило же, в соответствии с которым транзакты выбираются для продвижения по модели, определяется их положением в так называемых цепях. В настоящем разделе рассматриваются две цепи: текущих событий (ЦТС) и будущих событий (ЦБС), являющихся основными для понимания логики работы интерпретатора. Имеется только одна ЦТС и одна ЦБС.

ЦТС состоит из тех транзактов, для которых планируется их продвижение по одному или нескольким блокам модели в течение текущего значения модельного времени, а также тех транзактов, движение которых заблокировано ввиду условий в модели (например, движение транзакта может быть заблокировано ввиду

планирования его входа в блок SEIZE, когда требуемый прибор занят).

ЦБС состоит из таких транзактов, движение которых не планируется до наступления некоторого времени в будущем. Эти условия могут возникнуть только в двух случаях.

Транзакт попал в блок ADVANCE.

Транзакт должен войти в модель через блок GENERATE в более поздний момент времени.

Транзакты в ЦБС отсортированы в порядке запланированных будущих моментов времени. Например, транзакт, выход которого из блока ADVANCE запланирован на момент модельного времени 48, будет ближе к началу ЦБС, чем транзакт, выход которого из блока ADVANCE (того же или другого) запланирован на время 56.

GPSS изменяет состояние модели при просмотре ЦТС от начала к концу, транзакт за транзактом. При анализе каждого транзакта интерпретатор двигает его по модели до тех пор, пока не встретится одна из трех ситуаций, описанных в п.2.1.

Когда транзакт прекращает движение, интерпретатор выполняет одно из двух действий.

Продолжая просмотр ЦТС, выбирает следующий транзакт и пытается продвинуть его по модели.

Без продвижения таймера интерпретатор начинает просмотр ЦТС заново, т.е. возвращается к началу цепи, выбирает первый транзакт, двигает его по модели насколько это возможно и т.д.

Просмотр ЦТС начинается заново, если при своем движении только что остановленный транзакт прошел через блоки SEIZE или RELEASE (а также некоторых, пока еще не изученных, блоков). Причина возобновления просмотра ЦТС заключается в том, что, например, вход транзакта в блок RELEASE и освобождение прибора, возможно, снимет условия, блокирующие продвижение других транзактов в данный момент времени. Целью просмотра ЦТС заново и является обнаружение таких транзактов и продвижение их по модели.

Предположим теперь, что только что обработан транзакт, находящийся в конце ЦТС. Тогда интерпретатор проверяет транзакты от начала ЦБС. Он продвигает ТМВ к значению,

запланированному для движения первого транзакта в ЦБС. Этот транзакт переносится из ЦБС в ЦТС. Из ЦБС в ЦТС переносятся также все транзакты, движение которых можно возобновить в это новое значение модельного времени. Транзакты, переносимые из ЦБС, занимают в ЦТС положение в соответствии со своим уровнем приоритета (напомним, что уровень приоритета задается операндом E блока GENERATE и может быть изменен блоком PRIORITY, который будет рассмотрен ниже). Чем выше уровень приоритета, тем ближе к началу ЦТС располагается транзакт.

После того, как перенос из ЦБС завершен, интерпретатор опять приступает к просмотру ЦТС и т.д.

Рассмотрим, как интерпретатор GPSS использует ЦТС и ЦБС на примере моделирования 1. Для каждого транзакта в ЦТС или в ЦБС запоминается следующая информация: номер транзакта; уровень приоритета; время, на которое запланирована попытка войти в следующий блок; номер блока, в котором транзакт находится; номер следующего блока, в который делается попытка войти. Эту информацию будем записывать в виде пяти символов в указанном выше порядке. Например, запись [9.0.68.5.6] показывает, что транзакт с номером 9 и нулевым приоритетом пытается в момент времени, равный 68, выйти из блока 5 и войти в блок 6.

Предположим, что первые три значения интервалов времени между поступлениями транзактов в модель, полученные интерпретатором, определяются табл. 2.7, а первые два значения длительностей обслуживания задаются табл. 2.8.

В табл. 2.9 показаны состояния ЦТС и ЦБС в течение нескольких первых значений ТМВ. Рассмотрим ее подробнее.

Таблица 2.7.

N	Значение интервала
1	14
2	13
3	17

Таблица 2.8.

N	Значение интервала
1	13
2	12

Таблица 2.9.

N	ТМВ (Т)	ЦТС	ЦБС
1	До фазы ввода	Пусто	Пусто
2	После фазы ввода	Пусто	[1.0.14.0.1]
3	14	[1.0.T.0.1]	Пусто
4	14	Пусто	[2.0.27.0.1] [1.0.27.5.6]
5	27	[2.0.T.0.1] [1.0.T.5.6]	Пусто
6	27	Пусто	[2.0.39.5.6] [3.0.44.0.1]

Фаза ввода модели (от строки 1 к строке 2). Первым действием интерпретатора является фаза ввода модели. До фазы ввода ЦТС и ЦБС, естественно, пуста (строка 1). Во время фазы ввода интерпретатор заносит в счетчик числа завершений значение операнда A команды START и отыскивает блоки GENERATE. Как только встречается блок GENERATE, интерпретатор определяет время прихода первого транзакта в этот блок. Если задан операнд **C** , то время прихода устанавливается равным значению этого операнда. Если операнд C не задан, то разыгрывается время прихода в соответствии с распределением, определяемым операндами A и B. Далее интерпретатор выбирает транзакт из вершины пассивного буфера и помещает его в ЦБС с целью ввести в модель через блок GENERATE в полученный момент времени.

В нашем примере имеется один блок GENERATE. При его чтении на фазе ввода модели интерпретатором разыгрывается

время прихода первого транзакта, из пассивного буфера выбирается транзакт 1 и помещается в ЦБС с тем, чтобы ввести его в модель через блок GENERATE в момент времени, равный 14 (первое значение, табл.2.7).

На этом фаза ввода завершается. Состояния цепей после этой фазы определяются строкой 2 табл. 2.9. Заметим, что транзакт №1 еще не вошел ни в один блок модели, поэтому в его описания в ЦБС на месте номера текущего блока стоит "0". По завершении фазы ввода интерпретатор переходит к фазе коррекции ТМВ, т.е. к его приращению и переносу транзактов из ЦБС в ЦТС.

Первая коррекция таймера (от строки 2 к строке 3). Интерпретатор устанавливает ТМВ в значение 14 - значение времени движения первого транзакта в ЦБС - и перемещает этот транзакт из ЦБС в ЦТС. Поскольку в ЦБС транзактов, которые необходимо перенести в ЦТС, больше нет, то на этом фаза коррекции завершается. Состояния цепей после этой фазы даны в строке 3 табл.2.9. Отметим, что в ЦТС третьим элементом описания транзакта всегда является буква «Т», говорящая о том, что транзакты "хотели бы" войти в следующий блок при текущем ТМВ.

Первое выполнение фазы просмотра (от строки 3 к строке 4). Выбрав транзакт №1 из начала ЦТС, интерпретатор двигает его в блок №1 (GENERATE), затем проверяет, может ли он перейти в следующий блок. Следующим является блок QUEUE, который никогда не запрещает вход транзакта. Теперь, поскольку транзакт может покинуть блок GENERATE, то интерпретатор временно приостанавливает его движение и планирует приход следующего транзакта в блок GENERATE. Согласно табл. 2.7, время прихода следующего транзакта будет равно «Т+13», т.е. 27. Транзакт № 2 выбирается из вершины пассивного буфера и помещается в ЦБС. После этого интерпретатор вновь возвращается к транзакту №1 и продвигает его через блоки QUEUE, SEIZE, DEPART, ADVANCE. В блоке ADVANCE определяется время задержки транзакта (первое значение в табл.2.8). Транзакт № 1 выбирается из ЦТС и помещается в ЦБС, планируется его переход из блока 5 в блок 6 в момент времени, равный "Т+13", т.е. 27. Отметим, что он помещается в ЦБС за транзактом № 2, так как последний попал в ЦБС раньше.

Поскольку транзакт №1 прошел через блок SEIZE, то интерпретатор заново просматривает ЦТС. ЦТС пуста. Следовательно, необходима коррекция таймера.

Вторая коррекция таймера (от строки 4 к строке 5). Интерпретатор устанавливает ТМВ в значение 27 и переносит первый по порядку транзакт (транзакт № 2) из ЦБС в ЦТС. Время движения следующего транзакта также равно 27 (транзакт №1). Следовательно, и он переносится из ЦБС в ЦТС. На этом фаза коррекции завершается. Состояние цепей после нее даны в строке 5 табл. 2.9.

Второе выполнение фазы просмотра (от строки 5 к строке 6). Отметим, что в данном случае запланированы два события на один и тот же момент модельного времени (поступление в модель транзакта №2 и окончание обслуживания на приборе транзакта №1). Такие события называются одновременными, а момент модельного времени - временным узлом. События, вовлеченные в один временной узел, выполняются последовательно без продвижения ТМВ в том порядке, в каком оказались транзакты в ЦТС. Рассмотрим действия интерпретатора в данной ситуации.

Взяв транзакт №2 из начала ЦТС, интерпретатор двигает его в блок №1 (GENERATE). Затем, поскольку его можно переместить в блок №2 (QUEUE), обработка транзакта № 2 временно приостанавливается и разыгрывается время прихода следующего транзакта (T+I7, т.е. 44). Из пассивного буфера извлекается транзакт №3 и помещается в ЦБС (строка 6). Интерпретатор возобновляет движение транзакта №2, завершая его движение в блоке QUEUE, так как прибор занят.

Далее интерпретатор переходит к следующему тракзакту в ЦТС (№1) и, проводя его через блоки RELEAZE, TERMINATE, выводит из модели и возвращает в вершину пассивного буфера. Поскольку был пройден блок RELEAZE, то интерпретатор заново просматривает ЦТС, выбирает оставшийся в ней транзакт №2 и двигает его по модели до блока ADVANCE, где разыгрывается время задержки (второе значение в табл. 2.8). Затем этот транзакт переносится в ЦБС. Отметим, что в ЦБС он займет место перед

транзактом №3, так как значение времени его движения 39=T+12 меньше, чем значение времени движения транзакта №3.

ЦТС пуста, следовательно, следующим шагом является выполнение коррекции ТМВ. На этом мы закончим описание состояний цепей, показанных в табл.2.9.

2.11. Распределения вероятностей в GPSS

2.11.1. **Генераторы равномерно распределенных случайных чисел**. В GPSS/PC возможно обращение к семи генераторам случайных чисел с именами от RN1 до RN7 , с помощью которых можно получить случайное число, равномерно распределенное в интервале 0,…,0.999. В GPSS World количество генераторов случайных чисел не ограничено, а выдаваемое ими число находится в диапазоне 0,…,0.999999.

Генераторы работают независимо друг от друга, но если номера обращения к разным генераторам одинаковы, то и полученные случайные числа также будут одинаковы. Например, седьмое обращение к RN1 дает такое же случайное число, как и седьмое обращение к RN2 или любому другому генератору.

2.11.2. **Получение равномерных распределений**. Случайное число, равномерно распределенное на интервале A±B , вычисляется интерпретатором GPSS по формуле A-B+RN1·(2·B+1), где вместо RN1 подставляется случайное число, равномерно распределенное на [0,1) , получаемое при обращении к генератору RN1.

Так, если на фазе ввода интерпретатор находит блок GENERATE 18,6 (см. пример моделирования 1), то для планирования прихода первого транзакта через этот блок необходимо случайные числа выбрать из интервала 18±6. Пусть, например, при обращении к RN1 получено значение 0,194160. Тогда целой частью выражения 18-6+0,194160·(2·6+1) будет 14. Транзакт №1 выбирается из пассивного буфера и помещается в ЦБС с тем, чтобы в момент времени 14 он попал в блок 1.

2.11.3. **Определение дискретных функций в GPSS** . При определении функции распределения (ФР) дискретной с. в. в GPSS необходимо задать следующую информацию:

- имя функции (числовое или символическое);

- имя генератора случайных чисел (RNj), j=1,2,…,
- число различных значений, которые может принимать с. в.;
- значения с. в. и соответствующие значения ФР.

Запись этой информации рассмотрим на примере с. в., закон распределения которой задается табл. 2.10.

Таблица 2.10.

X_i	1	2	4	7	9	13
P_i	0,1	0,15	0,25	0,2	0,25	0,05

Пример определения функции в GPSS, описывающей данное распределение, приведен в табл. 2.11.

Таблица 2.11.

FUN1 FUNCTION RN1,D6
.1,1/.25,2/.5,4/.7,7/.95,9/1,13

В поле имени записывается имя функции, определяемое разработчиком (в примере – FUN1), затем в поле операций записывается слово FUNCTION, а в поле операндов - имя генератора случайных чисел (в примере – RN1) и после запятой - последовательность символов Dk, где D означает, что функция дискретная, а k определяет число различных значений, принимаемых с.в. (в примере - D6). Далее с новой строки, начиная с первой колонки, записываются в виде пар чисел суммарные частоты и соответствующие им значения с.в. Пары отделяются друг от друга символом «/». Если последовательность пар не может быть размещена в одной строке, то ее можно перенести на следующую строку. При этом строка продолжения должна начинаться с первой колонки, а в конце продолжаемой строки символ «/» не ставится.

Получение значения с.в. в соответствии с заданным распределением производится следующим образом. Вначале с помощью генератора, указанного в определении функции, получается случайное число. Затем интерпретатор, просматривая слева направо пары чисел, проверяет, является ли полученное случайное число по величине меньшим или равным значению

первого числа i-й пары, i= 1,2,... т.е. суммарной частоты. Если условие выполняется, то в качестве значения с.в. берется второе число i-й пары, i= 1,2,...

2.11.4. **Использование дискретных функций в блоках GENERATE и ADVANCE**. Если требуется, чтобы поступление транзактов в модель через блок GENERATE или длительность их задержки в блоке ADVANCE были отличны от детерминированного или равномерного, то необходимо:

- определить функцию, описывающую соответствующее распределение;
- в качестве операнда A в этих блоках указать ссылку на нужную функцию; операнд B при этом не задается.

Запись ссылки на функцию в операнде A блоков GENERATE и ADVANCE зависит от типа имени функции. Если было задано символическое имя, например, FUN1 , то операнд A записывается как FN$FUN1. Если же было задано числовое имя, например 1, то операнд A записывается как FN1.

2.11.5. **Определение непрерывных функций в GPSS** . Формально определение непрерывной функции в GPSS отличается от определения дискретной функции только тем, что вместо символа D записывается символ C. Расчет пар значений, входящих в определение функции, является в непрерывном случае существенно более сложной задачей. Желающие ознакомиться с методикой такого расчета могут воспользоваться книгой [4]. Так для того, чтобы задать экспоненциальное либо нормальное распределение, в качестве пар значений, входящих в определение функций, необходимо взять числа из соответствующих таблиц, приведенных нами в п.1.3. В приводимом ниже примере дано определение непрерывной функции GPSS (с числовым именем 1 и символическим именем XPDIS соответственно), используемой для розыгрыша чисел в соответствии с экспоненциальным распределением со средним значением, равным единице:

```
1       FUNCTION      RN1,C24
 0,0/.1,.104/.2,.222/.3,.355/.4,.509/.5,.69/.6,.915/.7,1.2/.75,1.38/.8,1.6
.84,1.83/.88,2.12/.9,2.3/.92,2.52/.94,2.81/.95,2.99/.96,3.2/.97,3.5
.98,3.9/.99,4.6/.995,5.3/.998,6.2/.999,7/.9998,8
```

Данная функция используется в блоках GENERATE и ADVANCE следующим образом:

- в качестве операнда A этих блоков используется среднее значение;

- в качестве операнда B используется ссылка на функцию, причем ссылка записывается в виде FN$ имя функции, если имя функции было задано в символьном виде либо в виде FNj, где j - числовое имя функции;

- в качестве используемого значения берется целая часть от произведения значений операндов A и B.

В заключение данного раздела отметим, что определение функции может быть сделано в любом месте модели, но обязательно до того, как используется ссылка на данную функцию.

2.11.6. **Моделирование вероятностных функций распределения в GPSS World.**

В GPSS World в библиотеку процедур включено 24 вероятностных распределения. При вызове вероятностного распределения требуется определить аргумент ***Stream*** (может быть выражением), который определяет номер генератора случайных чисел. При моделировании генераторы случайных чисел создаются по мере необходимости и их явное определение не обязательно. Большинство вероятностных распределений имеют некоторые параметры. Аргументы процедур, называемые обычно ***Locate, Scale*** **и** ***Shape,*** часто используются для этих целей. Аргумент ***Locate*** используется после построения применяемого распределения и прибавляется к нему. Это позволяет горизонтально перемещать функцию распределения по оси X. Аргумент ***Scale*** обычно меняет масштаб функции распределения, а ***Shape*** - ее форму.

Встроенная библиотека процедур содержит следующие вероятностные распределения [3]:

1) бета (Beta);
2) биномиальное (Binomial);
3) Вейбулла (Weibull);
4) дискретно-равномерное (Discrete Uniform);

5) гамма (Gamma);
6) геометрическое (Geometric);
7) Лапласа (Laplace);
8) логистическое (Logistic);
9) логлапласово (LogLaplace);
10) логлогистическое (LogLogistic);
11) логнормальное (LogNormal);
12) нормальное (Normal);
13) обратное Вейбулла (Inverse Weibull);
14) обратное Гаусса (Inverse Gaussian);
15) отрицательное биномиальное (Negative Binomial);
16) Парето (Pareto);
17) Пирсона типа V (Pearson Type V);
18) Пирсона типа VI (Pearson Type VI);
19) Пуассона (Poisson);
20) равномерное (Uniform);
21) треугольное (Triangular);
22) экспоненциальное (Exponential);
23) экстремального значения A (Extreme Value A);
24) экстремального значения B (Extreme Value B).

В качестве примера покажем, как для генерации потока транзактов можно использовать библиотечную процедуру экспоненциального распределения с параметром $\lambda = 0{,}25$ и использованием генератора случайных чисел RN1:

GENERATE (Exponential (1,0,(1/0.25)))

В данном примере цифра 1 задает номер генератора с.в., цифра 0 определяет смещение данного распределения, а выражение 1/0.25 задает его математическое ожидание.

Из всех приведенных распределений опишем те, которые наиболее часто используются на практике.

Логарифмически нормальное распределение. Логарифмически нормальное распределение (логнормальное) - это распределение случайной величины, натуральный логарифм которой нормально распределен. Это распределение пригодно для

моделирования мультипликативных процессов так же, как нормальное - для аддитивных.

С помощью центральной предельной теоремы можно показать, что произведение независимых положительных случайных величин стремится к логарифмически нормальной случайной величине.

Логнормальная случайная величина формируется под влиянием большого числа независимых факторов, причем каждый отдельный фактор оказывает равномерно незначительное и равновероятное по знаку влияние. Прирост каждого следующего фактора пропорционален уже достигнутому к этому времени значению исследуемой величины. То есть рассмотренный характер воздействия является мультипликативным.

Функция плотности логнормального распределения:

$$f_\eta(x) = \frac{1}{\sqrt{2\pi}\sigma(x-\lambda)} e^{\frac{(\ell n(x-\lambda)-\mu)^2}{2\sigma^2}}$$

Если $x > \lambda$, в противном случае $f_\eta(x) = 0$.

Если после логарифмирования каждого элемента некоторого набора данных этот трансформированный набор данных нормально распределен, то исходные данные логарифмически нормально распределены.

Это распределение используется при моделировании экономических, информационных, физических и биологических систем. Оно хорошо моделирует процессы в случае, когда значение наблюдаемой переменной является случайной долей от значения предыдущего наблюдения.

Примерами использования этого распределения могут быть:

1) размеры и вес частиц, образуемых при дроблении;
2) доход семьи;
3) зарплата работников;
4) долговечность изделия, работающего в режиме износа и старения;
5) размер банковского вклада;
6) длины слов в языке;

7) длины передаваемых сообщений.

Например, когда неизвестно распределение длины передаваемых сообщений, размера файлов или длины запроса к базе данных, то с большой вероятностью можно предположить логнормальное распределение для этих величин.

Для вызова логнормального распределения используется библиотечная процедура:

LOGNORMAL(Stream**,** Locate, Scale, Shape),

где ***Stream*** - номер генератора случайных чисел, автоматически преобразуется в целое число, которое должно быть больше или равно 1; Locate=λ, Scale=σ, Shape=μ. Все параметры обязательные.

Гамма-распределение является обобщенным распределением Эрланга для случая, когда число α суммируемых величин является нецелым. Гамма-распределенная величина имеет значения от 0 до +∞, то есть неотрицательна. Если α - целое, то это будет распределение Эрланга.

Функция распределения значительно изменяет свою форму при различных параметрах, что позволяет использовать это распределение для моделирования различных физических явлений.

Гамма-распределение можно интерпретировать как сумму квадратов нормально распределенных случайных величин, то есть как χ^2-распределение.

Таким образом, χ^2 -распределение, распределение Эрланга и экспоненциальное распределение являются частными случаями гамма-распределения .

Функция плотности гамма-распределения имеет вид:

$$f_\gamma(x) = \frac{\beta^{-\alpha}(x-\lambda)^{\alpha-1}e^{\frac{(x-\lambda)}{\beta}}}{\Gamma(\alpha)},\ x>\lambda$$

где $x \geq 0$ и $f_\gamma(x) = 0$, если $x < 0$

Для вызова гамма-распределения используется библиотечная процедура

GAMMA (Stream, Locate, Scale, Shape),

где ***Stream*** - номер генератора случайных чисел, автоматически преобразуется в целое число, которое должно быть больше или равно 1 ; Locate=λ, Scale=β, Shape=α. Все параметры обязательные.

Когда аргумент ***Shape*** равен 1, гамма-распределение вырождается в экспоненциальное. Это означает, что **GAMMA** (Stream, Locate, Scale, 1**)** имеет то же распределение, что и **EXPONENTIAL** (Stream, Locate, Scale).

Распределение Вейбулла. Это распределение используется при моделировании жизненного цикла сложного изделия или индивидуума.

Функция плотности распределения Вейбулла имеет вид:

$$f(x) = \frac{\alpha(x-\alpha)^{\alpha-1}}{\beta^{\alpha}} e^{-(\frac{x-\lambda}{\beta})^{\alpha}},$$

где $x>\alpha$, в противном случае $f(x)=0$.

Параметр α задает форму распределения, β - интенсивность отказов, λ - величину сдвига для определения местоположения распределения.

Для вызова распределения Вейбулла используется библио-течная процедура

WEIBULL (Stream, Locate, Scale, Shape),

где ***Stream*** номер генератора случайных чисел, автоматически преобразуется в целое число, которое должно быть больше или равно 1; Locate=λ, Scale=β, Shape=α. Все параметры обязательные.

2.12. Блоки ENTER, LEAVE. Оператор STORAGE

В языке GPSS существуют элементы, называемые «многоканальными устройствами» (МКУ), понятие которых практически соответствует понятию параллельно работающих приборов многоканальной системы в теории СМО. В дальней-

шем для краткости вместо термина «многоканальное устройство» будем использовать термин «устройство», число приборов в устройстве называть его емкостью, а число свободных приборов - свободной емкостью устройства. Использование устройства для моделирования работы одного из параллельно работающих приборов во многом аналогично использованию одного прибора. Элементами, требующими обслуживания в устройстве, являются транзакты. Устройство занимается и освобождается транзактом, проходящим через блоки ENTER и LEAVE соответственно.

Формат блоков:

ENTER A[,B]

LEAVE A[,B]

Таблица 2.12.

Операнд	Значение	Значение по умолчанию
A	Имя (символическое или числовое) устройства	Ошибка
B	Количество занимаемых одновременно устройств	1

Когда транзакт пытается войти в блок ENTER, осуществляется проверка, имеется ли свободная емкость в устройстве. Если устройство полностью занято, то транзакт ожидает появления свободной емкости в блоке, из которого он пытался войти в блок ENTER. Когда транзакт входит в блок ENTER, интерпретатором выполняются следующие действия:

- счетчик входов устройства увеличивается на единицу;
- счетчик текущего содержимого устройства увеличивается на единицу;
- свободная емкость устройства уменьшается на единицу.

Заметим, что эти действия во многом аналогичны действиям

интерпретатора, выполняемым при входе транзакта в блок QUEUE.

Блок LEAVE не запрещает вход транзакта. Причем устройства могут освобождаться транзактами, которые их не занимали, т.е. в блок LEAVE может войти транзакт, который до этого не проходил через блок ENTER. Этим устройства отличаются от приборов. Когда транзакт входит в блок LEAVE, то интерпретатор выполняет следующие действия:

- счетчик текущего содержимого устройства уменьшается на единицу;
- свободная емкость устройства увеличивается на единицу.

Эти действия аналогичны действиям интерпретатора при входе транзакта в блок DEPART.

Все используемые в модели устройства должны быть заранее описаны, т.е. должно быть определено количество однотипных устройств, входящих в многоканальное устройство. Для этого используется оператор STORAGE:

Таблица 2.13.

Поле	Информация в поле
Метка	Символическое имя устройства
Операция	STORAGE
Операнд А	Емкость устройства

Моделирование процесса обслуживания заявок на приборах в многолинейной СМО осуществляется последовательностью блоков ENTER – ADVANCE – LEAVE. В процессе моделирования по всем устройствам модели интерпретатором собирается стандартная статистика, которая распечатывается при завершении прогона модели в следующем виде:

Таблица 2.14.

STORAGE	CAP.	REMAIN	MIN	MAX	ENTR.	AVL.	AVE. C.	UTIL.
RPOOL	3	3	0	1	50	1	0,99	0,33

Поле **STORAGE** определяет имя или номер МКУ.

Поле **CAP.** определяет емкость МКУ, заданную оператором **STORAGE.**

Поле **REMAIN** определяет количество единиц свободной емкости МКУ в конце периода моделирования.

Поле **MIN** определяет минимальное количество используемой емкости МКУ за период моделирования.

Поле **MAX** определяет максимальное количество используемой емкости МКУ за период моделирования.

Поле **ENTRIES** определяет количество входов в МКУ за период моделирования.

Поле **AVL.** определяет состояние готовности МКУ в конце периода моделирования: 1 - МКУ готов, 0 - не готов.

Поле **AVE.C** определяет среднее значение занятой емкости за период моделирования.

Поле **UTIL.** определяет средний коэффициент использования всех устройств МКУ.

Для GPSS/PC статистику о работе МКУ можно наблюдать в окне МКУ, перейдя в это окно с помощью клавиш [ALT+S], а для GPSS World - в окне Storages Window.

В заключение отметим, что блоки ENTER и LEAVE являются блоками, при входе транзантов в которые интерпретатор начинает заново просмотр ЦТС.

2.13. Пример моделирования 2.

Постановка задачи. Построить модель работы порта, в который прибывают морские суда двух типов, и происходит их погрузка. В порту имеется два буксира, позволяющие осуществлять ввод кораблей в порт и их вывод на рейд. К первому типу судов относятся корабли малого тоннажа, которые требуют использования одного буксира. Корабли второго типа имеют большие размеры, и для их буксировки требуется два буксира. Из-за различия размеров судов в порту имеется два типа причалов. Время погрузки и разгрузки для кораблей двух типов также различно. Данная задача впервые рассмотрена в работе [3]. Исходные данные для нее приведены в таблице 2.15.

Таблица 2.15.

Значение	Тип корабля	
	1	2
Интервал прибытия, мин.	130±30	390±60
Время входа в порт, мин.	30±7	45±12
Количество причалов	6	3
Время погрузки-разгрузки, час	12±2	18±4
Время выхода из порта, мин.	20±5	35±10

Требуется оценить время ожидания входа в порт кораблями каждого типа, наблюдая за работой порта в течение 30-ти суток.

Распечатка модели и результатов моделирования приведена в приложениях 2.2.1 и 2.2.2 соответственно. Блоки программы снабжены подробными комментариями, что позволяет достаточно легко разобраться в логике программирования.

2.14. Блок TRANSFER

2.14.1. **Блок TRANSFER в режиме безусловной передачи.** Если возникает необходимость передать транзакт безусловным образом в блок, отличный от последующего, то это можно сделать, используя блок TRANSFER в режиме безусловной передачи. Значения операндов блока приведены в табл. 2.16.

Таблица 2.16

Операнд	Значение	Значение по умолчанию
A B	Не используется Символическое имя адресуемого блока	- Ошибка

В этом режиме операнд A не используется и вместо него ставится запятая. В качестве операнда B используется символическое имя блока, в который должен быть направлен транзакт.

Если блок, в который направляется транзакт, запрещает его вход, то транзакт остается в блоке TRANSFER , а с точки зрения цепей - в ЦТС.

2.14.2. **Блок TRANSFER в режиме условной передачи**. Если требуется передать транзакт в один из двух блоков в зависимости от условий, существующих в настоящий момент в модели, то используется блок TRANSFER в режиме условной передачи. Значения операндов блока приведены в табл. 2.17.

Таблица 2.17.

Операнд	Значение	Значение по умолчанию
A	BOTH	Ошибка
B	Символическое имя адресуемого блока	Следующий блок
C	Символическое имя адресуемого блока	Ошибка

Транзакт, вошедший в блок TRANSFER, работающий в режиме условной передачи, немедленно пытается перейти в блок с именем, указанным в операнде B блока TRANSFER. Если вход невозможен, то транзакт пытается войти в блок с именем, указанным в операнде C блока TRANSFER . Если и этот блок

отказывает транзакту во входе, то он остается в блоке TRANSFER, а с точки зрения цепей - в ЦТС. При каждом последующем просмотре ЦТС описанная выше процедура повторяется.

2.14.3. **Блок TRANSFER в режиме статистической передачи**. Если требуется передать транзакт в один из двух блоков случайным образом вне зависимости от условий, существующих в настоящий момент времени в модели, то используется блок TRANSFER в режиме статистической передачи. Значения операндов блока приведены в табл.2.18.

Таблица 2.18.

Операнд	Значение	Значение по умолчанию
A	Вероятность передачи в блок с символическим именем, указанным в операнде C	Ошибка
B	Символическое имя адресуемого блока	Следующий блок
C	Символическое имя адресуемого блока	Ошибка

Транзакт, вошедший в блок TRANSFER, работающий в режиме статистической передачи с вероятностью, заданной операндом A, направляется в блок, имя которого задано операндом C, а с дополнительной вероятностью в блок с именем, заданным операндом B. При задании вероятности используется не более трех цифр после символа «.» (десятичная точка). Например:

```
TRANSFER    .355,NEXT1,NEXT12
```

Если вход транзакта в выбранный блок запрещен, то он остается в блоке TRANSFER, а с точки зрения цепей - в ЦТС. При каждом последующем просмотре ЦТС делается новая попытка войти в выбранный блок.

2.15. Стандартные числовые атрибуты

В процессе моделирования интерпретатор GPSS автоматически собирает статистическую информацию по различным элементам, используемым в модели. Часть этой информации распечатывается в конце прогона модели. Такой информацией являются счетчики блоков, загрузка приборов и устройств, средняя длина очередей и т.д. Автоматически собираемая интерпретатором GPSS информация называется стандартными числовыми атрибутами (СЧА) и может быть использована в процессе прогона модели.

Доступ к СЧА осуществляется при использовании специальных обозначений этих атрибутов. Имя СЧА состоит из двух частей. Первая часть идентифицирует тип элемента (т.е. прибор, устройство и т.д.) и тип информации (т.е. свободную емкость, текущее содержимое устройства и т.д.). Вторая часть имени идентифицирует конкретный элемент группы. Если имя элемента задано в числовой форме, то полное имя СЧА записывается в форме СЧАj, где j - числовое имя элемента. Если же имя элемента задано в символическом виде, то полное имя СЧА записывается в форме СЧА$имя.

СЧА устройств и очередей. Список основных СЧА, относящихся к устройствам и очередям, приводятся в табл. 2.19.

Таблица 2.19.

Обозначение	Значение
Fj или F$имя	Показатель занятости устройства (0 - если не занято, 1 - если занято)
FCj или FC$имя	Число занятий устройства
FRj или FR$имя	Нагрузка устройства, выраженная в долях
FTj или FT$имя	Целая часть значения среднего времени задержки транзакта в устройстве
FVj или FV$имя	Флаг готовности устройства к использованию (1 -готово, О- не готово)
Rj или R$имя	Емкость незаполненной части МКУ

Sj или S$имя	Емкость заполненной части МКУ
SAj или SA$имя	Целая часть среднего заполнения МКУ
SCj или SC$имя	Счетчик числа входов в МКУ. (При каждом выполнении блока ENTER значение счетчика увеличивается на значение операнда B этого блока)
SMj или SM$имя	Максимально занятая емкость МКУ. Запоминает максимальное значение Sj
SRj или SR$имя	Нагрузка МКУ, выраженная в долях тысячи
STj или ST$имя	Целая часть среднего времени пребывания транзакта в МКУ
SVj или SV$имя	Флаг готовности МКУ к использованию (1 - готово, 0 - не готово)
Qj или Q$ имя	Текущее значение длины очереди (текущее содержимое)
QAj или QA$имя	Целая часть среднего значения длины очереди
QCj или QC$имя	Число входов в очередь. При каждом входе в блок QUEUE очереди значение QCj (QC$имя) увеличивается на значение операнда B, при каждом входе в блок DEPART очереди значение QCj (QC$имя) уменьшается на значение операнда B
QMj или QM$имя	Максимальное значение длины очереди (максимальное значение Qj (Q$ имя)
QTj или QT$имя	Целая часть среднего времени пребывания в очереди всех транзактов, которые входили в очередь (включая и те, которые не ждали - нулевые входы)
QXj или QX$имя	Целая часть среднего времени пребывания в очереди для транзактов, которые ждали в очереди (ненулевые входы)
QZj или QZ$имя	Число нулевых входов в очередь

СЧА блоков и системные СЧА. Блоки имеют два стандартных числовых атрибута (их подсчет ведется автоматически):

Wj (W$<метка блока>) - счетчик текущего содержимого блока с номером j (с меткой блока);

Nj (N$<метка блока>) - счетчик входов, т.е. общее число транзактов, вошедших в блок с момента последнего действия операторов RESET и CLEAR (от начала работы модели, если не было операторов RESET и CLEAR).

Например, W$BL1 - это число транзактов, которые находятся в блоке с меткой BL1; W$QP - число транзактов, вошедших в блок с меткой QP; W210 - значение текущего содержимого счетчика блока, который имеет 210-ю позицию в модели.

Важные системные СЧА:

C1 - текущее значение относительного модельного времени; автоматически изменяется интерпретатором и устанавливается в ноль при выполнении операторов CLEAR и RESET;

AC1 - текущее значение абсолютного модельного времени; автоматически изменяется интерпретатором и устанавливается в ноль при выполнении оператора CLEAR;

TG1 - текущее значение счетчика завершения;

PR - приоритет транзакта, обрабатываемого в данный момент;

M1 - время пребывания в модели транзакта, обрабатываемого интерпретатором в данный момент.

2.16. Параметры транзактов. Блок ASSIGN

Транзакты в моделях могут иметь до ста параметров. Множество параметров транзакта представляют собой набор СЧА. Они являются локальными переменными, посколько доступны только данному транзакту. В процессе перемещения транзакта в модели его параметры могут модифицироваться пользователем в соответствии с логикой моделирования.

Особенности параметров транзактов:

1. Доступ к параметрам транзактов осуществляется следующим образом:

P<номер> или P$<имя>,

где P - СЧА транзакта, определяющий его групповое имя, т.е. имя всех параметров транзакта.

2. Номера (имена) конкретных членов множества параметров задаются с помощью целых чисел 1, 2, ... (символьных имен). Например, P22 - это 22-ой параметр транзакта, P$COLOR - параметр с именем COLOR.

3. При входе транзакта в модель начальное значение всех его параметров устанавливается в ноль.

4. Значения параметров транзактов и их изменение определяет пользователь.

5. Значениями параметров транзактов могут быть любые числа (в системе GPSS/PC - только целые числа). Параметры могут приобретать отрицательные значения.

6. Транзакт может обращаться только к своим параметрам. Если необходимо получить доступ к параметрам других транзактов, то это можно сделать только через ячейки сохраняемых величин (речь об этом пойдет ниже).

7. Параметры можно использовать в качестве операндов блоков и в качестве аргументов функций.

8. Параметры позволяют организовать косвенную адресацию блоков. Это дает возможность агрегированного представления объектов моделирования в программе.

Приведем некоторые примеры использования параметров транзактов:

Пример 1. Предположим, что в программе имеются следующие операторы:

```
AAA     FUNCTION      P3, D3
-3,4/3,7/10,8

ADVANCE       FN$AAA,3
```

Это означает, что при входе транзакта в блок ADVANCE сначала будет произведен расчет функции AAA. Для того, чтобы посчитать ее значение, необходимо определить величину третьего параметра транзакта. Пусть она равна 4. Тогда соответствующее

значение функции AAA получим равным 8. Таким образом, время задержки будет равномерно распределено на интервале 8±3.

Пример 2. Если в программе будет предусмотрен следующий блок:

```
SEIZE        P$PRIB
```

то транзакт займет устройство, номер которого содержится в параметре транзакта с именем PRIB.

Для изменения значений параметров транзактов в GPSS используется блок ASSIGN. При входе транзакта в этот блок значения его параметров могут задаваться или изменяться.

Формат блока:

```
ASSIGN    A[+,–], B[,C]
```

Назначение операндов блока ASSIGN указано в табл. 2.20.

Таблица 2.20.

Операнд	Значение	Результат по умолчанию
A	Номер или имя модифицируемого или задаваемого параметра	Ошибка
B	Величина, используемая для модификации (число или СЧА)	Ошибка
C	Имя функции	Не используется

Блок ASSIGN может быть использован как в режиме замещения значения параметра (начальное значение всех параметров транзактов равно 0), так и в режиме увеличения и уменьшения. В режиме увеличения предшествующее значение параметра увеличивается на значение, стоящее в операнде B. В режиме уменьшения оно уменьшается на величину, стоящую в операнде B. Режимы увеличения и уменьшения определяются

введением соответственно знаков «плюс» и «минус» перед запятой, которая разделяет операнды A и B.

При использовании операнда C значение операнда B умножается на значение функции, указанной в операнде C. Параметр, заданный в операнде A, изменяется на величину полученного произведения (в режиме увеличения и уменьшения) или приобретает значение результата (в режиме замещения).

Пример 3.

Блок ASSIGN в режиме присваивания:

```
ASSIGN      MEST,36
```

Параметру транзакта с именем MEST присваивается значение 36.

```
ASSIGN      3,25
```

Параметру 3 присваивается значение 25.

```
ASSIGN      P4,FR$BB
```

Параметру транзакта с номером, записанным в параметре P4, присваивается значение величины загрузки устройства BB (оба операнда заданы косвенным образом).

Блок ASSIGN в режимах накопления:

```
ASSIGN      4+,Q5
```

Параметр 4 увеличивается на значение, равное текущей длине очереди 5.

2.17. Блок PRIORITY

При входе транзакта в модель уровень его приоритета определяется значением операнда F блока GENERATE (см. п.2.4). Изменение приоритета транзакта при движении его в модели может иметь важное значение с точки зрения расположения его в ЦТС, что, в свою очередь, определяет последовательность, в которой транзакты выбираются для возобновления движения в модели.

Изменение уровня приоритета транзакта происходит при его входе в блок PRIORITY.

Этот блок имеет единственный операнд A, определяющий новый уровень приоритета транзакта.

Блок PRIORITY никогда не запрещает вход транзакта. При входе транзакта в этот блок выполняются следующие действия:

- назначается новый уровень приоритета;
- транзакт в ЦТС занимает последнее место среди транзактов своего уровня приоритета;
- интерпретатор продолжает продвигать транзакт по модели до тех пор, пока есть такая возможность;
- когда транзакт дальше продвинуть не удается, интерпретатор начинает просмотр ЦТС заново.

2.18. Пример моделирования 3. Постановка задачи. Рассмотрим модель однолинейной СМО с накопителем конечной емкости r = 5. На систему поступают два потока заявок, интервалы между поступлениями которых распределены экспоненциально со средними значениями, равными соответственно $1/\lambda_1 = 18$ и $1/\lambda_2 = 15$. Длительности обслуживания заявок обоих потоков на приборе постоянны и равны 12 и 8 единиц времени соответственно. Заявки первого потока обладают относительным приоритетом при выборе на обслуживания. Моделирование системы необходимо завершить по истечении 500000 единиц модельного времени.

Распечатка модели и результатов моделирования приведена в приложениях 2.3.1 и 2.3.2 соответственно.

2.19. Переменные

Интерпретатор GPSS сконструирован так, что ряд величин, таких, как загрузка прибора, среднее время пребывания в очереди и т.п., вычисляются без каких-либо указаний со стороны пользователя. Однако многие вычисления зависят от контекста конкретной модели и не могут быть выполнены автоматически. Для производства различного рода вычислений в GPSS используются переменные, которые будут рассмотрены в этом разделе.

2.19.1. Арифметические переменные

Арифметическая переменная в GPSS является СЧА. Задается она с помощью оператора VARIABLE. В поле имени

оператора указывается символическое либо числовое имя переменной. В поле операндов задается арифметическое выражение, определяющее правило вычисления значений переменной. Если выражение слишком длинное, оно должно быть разбито на несколько выражений и определено с помощью нескольких арифметических переменных. Арифметическое выражение представляет собой набор данных, связанных арифметическими операциями.

В GPSS допустимы следующие арифметические операции:

\+ - сложение;

\- - вычитание;

\# - умножение;

/ - деление нацело: дробная часть от деления отбрасывается;

@ - деление по модулю: целая часть от деления отбрасывается, а результатом деления является остаток (пример: 15@4=3);

^ - возведение в степень.

В арифметических выражениях операции умножения, деления нацело, деления по модулю, возведения в степень предшествуют сложению и вычитанию. Для изменения порядка следования операций в выражениях можно использовать скобки в соответствии с обычными правилами арифметики.

Данные в выражениях могут определяться как прямо, так и косвенно. При прямом задании данные принимают форму целых констант. Косвенно заданные данные представляются как СЧА. Так как арифметические переменные сами являются СЧА, то в арифметические выражения могут быть включены ссылки на другие арифметические переменные.

Ссылкой на арифметическую переменную является либо Vj, либо V$имя, где j -номер переменной, если она было задана этим номером, а "имя" - символическое имя в случае символического задания.

Ниже приведены два примера арифметических переменных:

```
BUFF        VARIABLE        Q$QUE1+Q2
```

```
1          VARIABLE          (V$MARK*2+P2)/4
```

Переменная BUFF определяется как сумма текущего содержимого очередей QUE1 и 2. Переменная 1 принимает значение, равное целой части от деления на 4 выражения в скобках. Выражение в скобках, в свою очередь, есть сумма удвоенного значения арифметической переменной MARK и значения второго параметра транзакта. Здесь необходимо заметить, что арифметическая переменная вычисляется только тогда, когда транзакт входит в блок, в котором величины операнда некоторым образом зависят от этой переменной.

2.19.2. Арифметические переменные с плавающей точкой

Арифметические переменные с плавающей точкой аналогичны рассмотренным арифметическим переменным, за исключением того, что все операции над операндами выражений переменных с плавающей точкой выполняются без преобразования операндов и промежуточных результатов в целые значения. Лишь окончательный результат вычисления преобразуется в целое число.

Формат операторов описания арифметических переменных с плавающей точкой идентичен рассмотренному выше формату операндов описания арифметических переменных за исключением того, что в поле операции записывается слово FVARIABLE. Правила написания операторов те же, что и для арифметических переменных. Арифметическая переменная и переменная с плавающей точкой не могут иметь одинаковые номера. Если они имеют одинаковые номера, то при вычислении используется более позднее из двух описаний.

Различие результатов, полученных при вычислении с плавающей точкой и фиксированной, можно увидеть из такого примера:

```
FLOAT          FVARIABLE    10#(11 /3)
FIXED          VARIABLE     10#(11/3)
```

Значение переменной FLOAT равно 36, так как константа 10 умножается на 3,67 и от результата 36,7 взята целая часть. Переменная FIXED равна 30, так как результат промежуточной операции деления будет округлен до 3.

2.19.3. Булевы переменные

Булевы переменные позволяют принимать решения в зависимости от значений СЧА и состояния объектов GPSS, используя для этого только одно выражение.

Булевы переменные - это логические выражения, состоящие из различных СЧА и (или) других булевых переменных. В булевой переменной проверяется одно или несколько логических условий. Результатом проверки есть единица (истина), если условия выполняются, и ноль (ложь) - в противном случае.

При описании булевых переменных используются три типа операторов: логические, булевы и операторы отношений.

Логические операторы связаны с такими ресурсами, как устройства, МКУ и логические ключи. Они используются для определения состояния данных объектов. Логические операторы, используемые в GPSS, представлены в табл. 2.21

Таблица 2.21

Логические операторы	Значение оператора, отражающее состояние ресурса
FVj или Fj	Равно 1 , если устройство j занято или обслуживает прерывание, в противном случае 0
FNVj	Равно 1 , если устройство j не занято и не обслуживает прерывание, в противном случае - 0
Ij	Равно 1, если устройство j обслуживает прерывание, в противном случае - 0
NIj	Равно 1, если устройство j не обслуживает прерывание, иначе - 0
NUj	Равно 1, если устройство j не используется, в противном случае - 0
Uj	Равно I, если устройство j используется, в противном случае - 0

SFj	Равно 1 , если многоканальное устройство j заполнено, иначе - 0
SNFj	Равно 1, если МКУj не заполнено, иначе -0
SEj	Равно 1, если МКУ j пусто, иначе - 0
SNEj	Равно 1 , если МКУ j не пусто, иначе - 0
SVj	Равно 1 , если МКУ j находится в состоянии использования, в противном случае - 0
SNVj	Равно 1 , если МКУ j не используется, в противном случае - 0
LRj	Равно 1 , если логический ключ j выключен, иначе -0
LSj	Равно 1 , если логический ключ j включен, иначе - 0

Операторы отношения выполняют алгебраическое сравнение операндов. Операндами могут быть константы или стандартные числовые атрибуты. Все операторы отношений записываются в кавычках:

"G" (Greater) - больше;

"L" (Less) — меньше;

"E" (Equal) - равно;

"NE" (Not Equal) - не равно;

"LE" (Less than or Equal) - меньше или равно;

"GE" (Greater than or Equal) - больше или равно.

Есть два булевых оператора: "OR" - оператор «или», и "AND" - оператор «и». Оператор «или» проверяет, выполняется ли хотя бы одно из проверяемых условий. Оператор «и» требует выполнения обоих условий.

2.20. Оператор INITIAL и блок SAVEVALUE

В GPSS пользователь имеет возможность закреплять постоянные области памяти для хранения некоторых числовых

величин. Такие величины называются сохраняемыми и относятся к разряду СЧА. На протяжении всего процесса моделирования сохраняемые величины (СВ) никуда не исчезают и изменяются только при прямом указании пользователя.

В GPSS возможно как одномерное, так и двумерное представление СВ. В дальнейшем одномерные СВ будем называть просто СВ, опуская слово "одномерные". Их рассмотрению и посвятим настоящий раздел. Двухмерные СВ называются матричными СВ. Они будут рассмотрены в следующем разделе.

СВ должны быть снабжены числовыми или символическими именами. Ссылками на сохраняемые величины являются Xj (или X$имя) - где j - номер СВ в случае ее числового задания, а '"имя" - символическое имя СВ в случае ее символического задания.

Начальные значения СВ задаются с помощью оператора INITIAL. Формат оператора INITIAL представлен в следующей таблице:

Таблица 2.22.

Поле	Информация в поле
Метка	Не используется
Операция	INITIAL
Операнд A	Имя сохраняемой величины
Операнд B	Начальное значение

Значение СВ изменяется при входе транзакта в блок SAVEVALUE. Этот блок имеет два операнда, назначение которых описано в табл.2.23.

Таблица 2.23.

Операнд	Значение	Значение по умолчанию
A	Номер или символическое имя СВ	Ошибка
B	Величина, которую следует сохранить	Ошибка

Как только транзакт входит в блок SAVEVALUE, величина операнда B становится новым значением СВ с номером (символическим именем), записанным в операнде A.

Блок SAVEVALUE можно использовать не только для замещения одного значения СВ другим, но и для его накопления либо уменьшения. В режиме накопления значение СВ увеличивается, а в режиме уменьшения - уменьшается на величину, равную значению операнда B . Режимы накопления и уменьшения определяются, введением соответственно знака «+» или «-» перед запятой, разделяющей операнды A и B .

Ниже приведены примеры использования блока SAVEVALUE:

```
SAVEVALUE        P2,Q$QUE
SAVEVALUE        WORK+,V2
```

В первом примере СВ, номер которой записан во втором параметре транзакта, вошедшего к блок SAVEVALUE присваивается значение, равное текущему содержимому очереди с именем QUE.

Во втором примере при входе транзакта в блок SAVEVALUE происходит подсчет значения переменной с номером 2, и получаемая величина прибавляется к значению СВ с именем WORK.

2.21. Оператор MATRIX и блок MSAVEVALUE

В отличие от одномерных СВ матричные СВ, используемые в модели должны быть описаны до начала моделирования. Это делается путем задания каждой матрицы оператором MATRIX . В поле имени этого оператора помещается символическое либо числовое имя задаваемой матрицы. В поле операндов операнд A не используется, а на его месте ставится «,». Это делается для совместимости с более ранними версиями GPSS.

Операнды B и C являются целыми числами и задают количество строк и столбцов матрицы соответственно. Оператор MATRIX должен быть помещен в модели перед любыми ссылками на рассматриваемую матрицу.

При ссылке на (i,j)-ый элемент матрицы употребляют символы MX, за которыми следует числовое либо символической имя матрицы и символы (i,j). Например, запись MX2(1,4) означает обращение к элементу матрицы 2, стоящему на пересечении первой строки и четвертого столбца. Запись MX$PM(4,5) является ссылкой на элемент, стоящий в четвертой строке и пятом столбце матрицы с символическим именем PM.

Номера строк и столбцов могут задаваться косвенно. Например, MX5(P3,P4) - ссылка на элемент матрицы 5. Номера строки и столбца равны соответственно значениям третьего и четвертого параметров транзакта, вошедшего в блок, ссылающийся на матрицу 5.

Перед началом моделирования все значения элементов матрицы устанавливаются равными нулю. По желанию пользователя некоторым элементам матриц могут быть присвоены ненулевые начальные значения с помощью оператора INITIAL. В качестве операнда A указывается имя матрицы и номер соответствующего элемента, а в качестве операнда B - начальное значение этого элемента. Например, запись

INITIAL MX2(1,2),5

означает, что элементу (1,2) матрицы 2 присвоено начальное значение 5.

Значение одного элемента матрицы изменяется при прохождении транзакта через блок MSAVEVALUE. Этот блок имеет четыре операнда, назначение которых описано в табл. 2.24.

Таблица 2.24.

Операнд	Значение	Значение по умолчанию
A	Имя матрицы (числовое или символическое)	Ошибка
B	Номер строки	Ошибка
C	Номер столбца	Ошибка
D	Величина, используемая при изменении элемента матрицы	Ошибка

Блок MSAVEVALUE может быть использован как в режиме замещения, так и в режиме накопления либо уменьшения. В режиме накопления значение элемента матрицы увеличивается, а в режиме уменьшения - уменьшается на величину, равную значению операнда D. Режимы накопления и уменьшения задаются помещением соответственно знака «-» или «+» перед запятой, разделяющей операнды A и B.

Ниже приведены примеры, показывающие использование блока MSAVEVALUE:

```
MSAVEVLUE        ALFA,P1,P2,Q 1
MSAVEVALUE       3+,l,P8,V5
```

В первом примере элементу матрицы ALFA , стоящему на пересечении строки и столбца с номерами, равными величинам первого и второго параметров транзакта соответственно, присваивается значение, равное текущему содержимому очереди 1. Во втором примере при входе транзакта в блок MSAVEVALUE происходит подсчет значения переменной 5. Полученная величина прибавляется к значению элемента матрицы 3, расположенному в первой строке и столбце, номер которого равен величине восьмого параметра транзакта.

Итак, мы рассмотрели вопросы, связанные с определением сохраняемых величин и матричных сохраняемых величин. Теперь кратко остановимся на некоторых аспектах их использования в моделях. Во-первых, с помощью СВ удобно вводить исходные параметры моделируемой системы. В рассмотренных ранее примерах моделирования исходные данные вводились непосредственно в качестве операндов блоков либо в определении функций, что создавало очевидные неудобства в тех случаях, когда одну и ту же модель необходимо использовать при различных начальных значениях исходных параметров. Во-вторых, наличие постоянных областей памяти позволяет записывать в них информацию, доступную любым транзактам. Последнее, в свою очередь, позволяет транзактам влиять друг на друга. Так, например, параметру одного транзакта может быть присвоено значение параметра другого транзакта, если последнее

было предварительно записано в память в качестве некоторой сохраняемой величины.

2.22. Блок TEST

Блок TEST служит для сравнения двух СЧА. Блок TEST имеет вспомогательный оператор X , называемый оператором отношения, и три операнда, назначение которых описано в табл. 2.25.

Таблица 2.25

Операнд	Значение	Значение по умолчанию
A	Имя первого СЧА	Ошибка
B	Имя второго СЧА	Ошибка
C	Необязательный. Имя блока, в которой переходит проверяющий транзакт, если результат проверки отрицательный	Проверка производится в режиме отказа
Оператор X	G -A больше B?	Ошибка
	GE - A больше или равно B?	
	E - A равно B?	
	NE - A не равно B?	
	LE - A меньше или равно B?	
	L - A меньше B?	

Оператор X задает способ сравнения стандартных числовых атрибутов, указанных в операндах A и B. Как показано в табл. 2.25, X может быть одним из следующих

буквенных символов: G, GE, E, NE, LE, L который помещается в поле оператора непосредственно за блоком TEST.

Операнд C является необязательным. Если операнд C отсутствует, то блок TEST работает в режиме отказа, т.е. вход в блок запрещается в том случае, если ответ на вопрос, подразумеваемый оператором отношения, является отрицательным. При запрещении входа транзакт остается в предыдущем блоке и возобновляет попытку войти в блок TEST всякий раз, когда просматривается цепь текущих событий.

Если операнд C используется, то блок TEST работает в режиме условной передачи. В этом случае транзакт, поступивший в блок TEST переходит в следующий за ним блок, если ответ на подразумеваемый вопрос является положительным, и переходит в блок, указанный в операнде C , если ответ отрицательный.

Ниже приведены примеры использования блока TEST:

```
TEST NE          P2,V5
TEST G           MX4(V2,P1),Q$QUE,AAA1
```

В первом примере транзакт будет пропущен через блок TEST в том случае, если значение его второго параметра будет отлично от значения переменной 5. В противном случае транзакт будет оставаться в предыдущем блоке до тех пор, пока ответ на поставленный в блоке TEST вопрос не станет положительным.

Во втором примере блок TEST используется в режиме условной передачи. Это означает, что транзакт перейдет в блок с меткой AAA1, если величина элемента матрицы 4 не будет превышать текущего содержимого очереди с именем QUE. Номер строки и номер столбца, на пересечении которых расположен данный элемент матрицы, равны соответственно значению переменной 2 и значению первого параметра транзакта. В том случае, если проверка дает положительный ответ, то транзакт перейдет в блок, следующий за блоком TEST.

2.23. Блок SPLIT

До сих пор ввод транзактов в модель осуществлялся только с использованием блока GENERATE. В GPSS предусмотрена еще одна возможность ввода транзактов при помощи блока SPLIT. При

входе транзакта в этот блок из пассивного буфера извлекаются несколько транзактов и немедленно помещаются в ЦТС. Таким образом, транзакт как бы имеет способность расщепляться. Входящий в блок SPLIT транзакт будем называть родителем, а вводимые дополнительные транзакты - потомками. Операнды блока SPLIT описаны в табл.2.26. Величина операнда A задает число потомков, вводимых в модель. Операнд B задает имя блока, в который будут направлены потомки. Родитель направляется в следующий по порядку блок. Операнд C является необязательным. Он указывает номер параметра, в который будут записаны порядковые номера родителя и его потомков со смещением

Таблица 2.26.

Операнд	Значение	Значение по умолчанию
A	Число дополнительных транзактов, вводимых в модель	Ошибка
B	Имя блока, куда будут направлены дополнительные транзакты	Ошибка
C	Номер параметра упорядочивания	Упорядочивания не просходит
D	Число параметров, которое должен иметь каждый потомок	Каждый потомок имеет такое же число параметров, что и родитель

на первоначальную величину этого параметра. Для пояснения положим, что значение операнда C равно 5, а 11 - величина 5-го параметра транзакта - родителя до входа в блок SPLIT . Тогда после выхода из блока SPLIT величина 5-го параметра транзакта - родителя станет равной 12. Величина 5-го параметра первого

вводимого в модель транзакта - потомка примет значение 13, второго - 14 и т.д.

Операнд D является необязательным. Он используется в том случае, когда необходимо увеличить либо уменьшить число параметров в транзактах - потомках. При этом заметим, что потомки ничем не отличаются от родителя. Они имеют тот же уровень приоритета и те же параметры, что и их родитель. Таким образом, если потомок имеет большее число параметров, чем его родитель, то дополнительным параметрам присваиваются нулевые значения. Если же у потомка меньше параметров, то их значения будут совпадать со значениями тех же параметров родителя.

Ниже приведен пример использования блока SPLIT:

```
SPLIT          X$JOB,AAA1,3,10
```

В данном примере к транзакту, вошедшему в блок SPLIT, добавляется число потомков, равное значению СВ JOB. При этом потомки переходят в блок с меткой AAA1. В третий параметр трзнзактов записываются их порядковые номера, а число параметров в транзактах - потомках равно 10.

2.24. Оператор TABLE и блок TABULATE.

Часто возникает необходимость проанализировать поведение некоторых с.в., значения которых подсчитываются в процессе моделирования. Для автоматизации процесса вычисления ряда характеристик с.в. в GPSS используются таблицы.

В модели может быть несколько таблиц. Каждая таблица должна быть определена в начале моделирования с помощью оператора TABLE. В поле имени этого оператора помещается числовое или символическое имя таблицы. Операнды оператора TABLE описаны в табл.2.27.

Таблица 2.27

Операнд	Значение	Значение по умолчанию
A	Имя случайной величины, значение которой должно учитываться в таблице. Операнд должен быть именем СЧА	Ошибка
B	Первое граничное значение	Ошибка
C	Ширина всех промежуточных интервалов	Ошибка
D	Общее число интервалов таблицы, включая интервалы слева и справа от граничных значений	Ошибка

Значения случайной величины отмечаются в таблице в те моменты, когда транзакты входят в блок TABULATE. Единственный операнд этого блока задает имя соответствующей таблицы. Больше никакой информации в блоке TABULATE не указывается, так как все необходимые сведения о предстоящем табулировании содержатся в операторе TABLE, описывающем данную таблицу.

Совокупность всех значений с.в. образуют выборку, для которой интерпретатор при прохождении транзакта через блок TABULATE определяет: среднее значение; стандартное отклонение (корень из дисперсии); сумму всех значений с.в., общее число элементов выборки, попадающих в каждый из интервалов, установленных в операторе TABLE; процентное соотношение числа значений выборки, попадающих в различные интервалы; отклонение значения с.в. от своего среднего значения и ряд других характеристик, которые не имеют для нас принципиального значения, и останавливаться на них мы не будем.

Ниже приведен пример использования оператора TABLE и блока TABULATE:

```
TAB1    TABLE       V$TIME,0,2,30
        TABULATE    TAB1
```

При входе транзакта в блок TABULATE в таблице с именем TAB1 будет табулироваться значение переменной с именем TIME. Граничные значения равны 0 и 30, а ширина промежуточных интервалов между граничными значениями равна 2.

2.25. Блок MARK

Часто требуется знать, сколько времени проходит при продвижении транзакта между двумя точками модели. С этой целью необходимо при попадании транзакта в первую интересующую нас точку сделать отметку абсолютного времени по действующему таймеру. Затем, когда транзакт придет во вторую интересующую нас точку, необходимо подсчитать интервал времени между значением, отмеченным в первой точке , и текущим значением таймера.

Первое действие выполняется при входе транзакта в блок MARK (отметить), единственный операнд которого задает прямо либо косвенно параметр транзакта, в который необходимо занести указанное время. При входе во вторую точку длина интересующего нас интервала времени может быть получена в виде стандартного числового атрибута MPj , где j- номер параметра, указанный в соответствующем блоке MARK.

Ниже приведен пример использования блока MARK:

```
MARK        2
…
TEST  L     MP2,V$TIME,AAA1
```

При входе транзакта в этот блок во второй параметр транзакта заносится значение абсолютного времени по действующему таймеру. Затем при входе в блок TEST, работающий в режиме условной передачи, сравнивается значение длины интервала времени, прошедшего с момента выхода транзакта из блока MARK, со значением переменной TIME.

2.26. Пример моделирования 4

Постановка задачи. Рассмотрим модель однолинейной СМО конечной емкости. На систему поступает поток основных заявок. Интервалы времени между поступлениями основных заявок также как и длительности их обслуживания имеют гиперэкспоненциальное распределение. В том случае, если система освобождается, на обслуживание немедленно поступает фоновая заявка. Длительность обслуживания фоновой заявки также имеет гиперэкспоненциальное распределение. Таким образом, прибор никогда не простаивает. Он занят либо основными, либо фоновыми заявками. Причем ни основные, ни фоновые заявки не прерывают обслуживание друг друга.

Для данной СМО необходимо построить модель, имитирующую ее работу в течение 500000 единиц модельного времени. При этом основные исходные данные должны быть заданы в виде сохраняемых величин (скалярных и матричных). Параметры всех гиперэкспоненциальных распределений должны быть различны.

Необходимо получить основные характеристики системы для следующих исходных данных:

- емкость накопителя r=5;
- вектор вероятностей распределения по фазам генерации основных заявок α=(0.-2;0.5;0.3);
- вектор средних длительностей пребывания основных заявок на фазах генерации λ=(2;3;1);
- вектор вероятностей распределения по фазам обслуживания основных заявок β=(0.3;0.4;0.3);
- вектор средних длительностей пребывания основных заявок на фазах обслуживания μ=(1;2;3);
- вектор вероятностей распределения по фазам обслуживания фоновых заявок γ=(0.4;0.6);
- вектор средних длительностей пребывания фоновых заявок на фазах обслуживания σ=(1;2).

Распечатка модели и результатов моделирования приведена в приложениях 2.4.1 и 2.4.2 соответственно.

2.27. Блоки LINK и UNLINK.

В GPSS имеется возможность временно извлекать транзакты из ЦТС и размещать их в так называемых цепях пользователя (ЦП). Через определенное время транзакты можно вернуть в ЦТС в том порядке, который определяет сам пользователь. Это обстоятельство позволяет изменять порядок следования событий, который до сих пор определялся уровнем приоритета и хронологической последовательностью поступления транзактов в модель.

Транзакт помещается в ЦП при входе в блок LINK . Наличие этого блока в модели определяет и само существование ЦП. Операнды блока LINK описаны в таблице 2.28.

Таблица 2.28

Операнд	Значение	Значение по умолчанию
A	Имя ЦП (числовое или символическое)	Ошибка
B	Определяет место в ЦП, на которое следует поместить транзакт: FIFO – встать в конец цепи; LIFO – встать в начало цепи; Pj - встать непосредственно перед транзактами с большей величиной j-го параметра	Ошибка

Операнд A задает числовое либо символическое имя ЦП, а операнд B - место в ЦП, на которое помещается транзакт. Если операнд A задан выражением FIFO, то транзакт помещается в конец цепи пользователя. Если операнд B задан выражением LIFO, то транзакт помещается в начало ЦП. Если же в операнде B стоит параметр Pj, то транзакты размещается в порядке возрастания их j-го параметра.

Возвращение транзактов в цепь текущих событий производится путем ввода некоторого транзакта (инициатора) в блок UNLINK. Операнды блока UNLINK описаны в табл. 2.29.

Таблица 2.29

Операнд	Значение	Значение по умолчанию
A	Имя ЦП (числовое или символическое)	Ошибка
B	Имя блока, в который переходят выведенные из ЦП транзакты	Ошибка
C	Число выводимых транзактов (константа, СЧА либо символы ALL)	Ошибка
D	Необязательный операнд. Определяет условия вывода транзактов: BACK-транзакты выводятся с конца ЦП; любой СЧА – из ЦП выводятся транзакты, для которых величина j-го параметра совпадает с величиной j-го параметра транзакта, инициирующего вывод, где j-значение СЧА в операнде D.	Транзакт выводится с начала ЦП

При входе транзакта, инициирующего вывод, в блок UNLINK из ЦП с именем, указанным в операнде A, выводится такое количество транзактов, каково значение операнда C. Если операнд C задан символами ALL, то из ЦП выводятся все находящиеся там транзакты. Выведенные транзакты переходят в блок, имя которого указано в операнде B.

Порядок вывода транзактов определяется операндом D. Если этот операнд не используется, то вывод производится по порядку, начиная с начала ЦП. Если в операнде D указан символ BACK, то

вывод производится с конца ЦП. Если операнд D задан в виде некоторого СЧА (в том числе константы), то сначала подсчитывается значение этого атрибута. Предположим, что результатом подсчета явилось некоторое число j. Тогда из ЦП выводятся лишь те транзакты, для которых значение j-го параметра совпадает со значением j-го параметра транзакта, инициирующего вывод. При этом вывод производится по порядку, начиная с начала ЦП, до тех пор, пока не будет выведено столько транзактов, сколько указано в операнде C, либо пока не будет достигнут конец цепи.

Ниже приведен пример использования блоков LINK и UNLINK.

```
LINK        P5,LIFO
UNLINK    BUFF,BYBY,V7,BACK
```

При входе транзакта в блок LINK он помещается в начало ЦП с номером, равным значению его пятого параметра.

При входе транзакта в блок UNLINK прежде всего подсчитывается значение переменной 7, которое определяет количество выводимых транзактов. Затем из ЦП с именем BUFF выводится указанное количество транзактов с последующей их передачей в блок с именем BYBY. При этом вывод идет от конца ЦП к ее началу.

2.28. Пример моделирования 5

Постановка задачи. Построим имитационную модель однопотоковой m-канальной экспоненциальной СМО с накопителем емкости r, $r>\infty$ (рис.2.1)

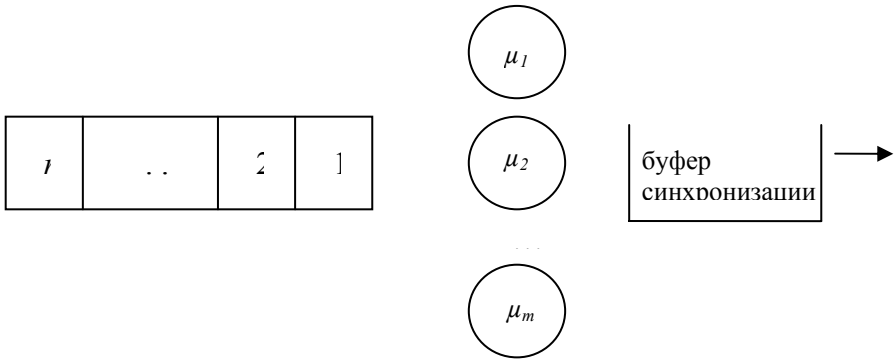

Рис.2.1

Поступающие в СМО заявки состоят из заданий, число которых не превосходит m. Задание *i* (*i* -задание) присутствует в заявке с вероятностью α_i , $0<\alpha_i<1$, и должно быть обслужено на приборе с номером *i*, $i=1,2,...,m$. По мере окончания обслуживания задания данной заявки накапливаются в буфере синхронизации. Заявка считается обслуженной, когда обслужены все составляющие ее задания. В момент окончания обслуживания заявки буфер синхронизации опустошается, и заявка покидает систему. Обслуживание заданий следующей заявки начинается только после окончания обслуживания всех заданий предыдущей заявки.

Предполагается, что поступающий в систему поток заявок является пуассоновским с параметром λ. Время обслуживания заданий на приборе распределено по экспоненциальному закону с параметром μ_i , $i=1,2,...,m$.

Описанная выше система известна в литературе как экспоненциальная СМО с синхронным циркулярным обслуживанием.

Для данной СМО необходимо построить модель при следующих исходных данных:

- количество приборов m=3;
- емкость накопителя r=3;

- средняя длина интервала между поступлениями заявок $1/\lambda=10$;
- вектор вероятностей присутствия заданий в заявке $\alpha=(0,8;0,9;0,95)$;
- вектор средних длительностей выполнения заданий $\mu^{-1}=(5;4;4)$.

Моделирование системы заканчивается после обслуживания 15000 заявок.

2.29. Блоки PREEMPT и RETURN

GPSS позволяет достаточно просто моделировать процедуру захвата прибора в то время, когда прибор занят обслуживанием другого транзакта. Такая возможность достигается при входе транзакта в блок PREEMPT. Возвращение захваченного прибора производится с помощью блока RETURN. Операнды блоков PREEMPT и RETURN описаны в табл. 2.30 и 2.31 соответственно.

Таблица 2.30

<table>
<tr><th>Операнд</th><th colspan="2">Значение</th><th>Значение по умолчанию</th></tr>
<tr><td>A</td><td colspan="2">Имя прибора (числовое или символическое), подлежащего захвату</td><td>Ошибка</td></tr>
<tr><td rowspan="3">B</td><td colspan="2">Необязательный операнд; указывает на условие, при котором захват разрешен:</td><td rowspan="3">Произойдет захват</td></tr>
<tr><td>Операнд B</td><td>Условия захвата</td></tr>
<tr><td>не используется

PR</td><td>произойдет захват

захват произойдет, если приоритет захватчика выше</td></tr>
</table>

<table>
<tr><td></td><td></td><td>приоритета обслуживаемого транзакта</td><td></td></tr>
<tr><td>C</td><td colspan="2">Необязательный операнд; имя блока, в который будет помещен прерванный транзакт</td><td>Транзакт будет выведен из ЦТС и помещен в цепь прерывания</td></tr>
<tr><td>D</td><td colspan="2">Необязательный операнд; задает номер параметра прерванного транзакта, в который записывается значение времени, оставшееся транзакту до окончания его обслуживания</td><td>Время не записывается</td></tr>
<tr><td rowspan="3">E</td><td colspan="2">Необязательный операнд; указывает на возможность автоматического восстановления прерванного транзакта на приборе</td><td rowspan="3">Транзакт сохраняет право восстановления на приборе</td></tr>
<tr><td>Операнд E</td><td>Значение</td></tr>
<tr><td>RE

не используется</td><td>Транзакт теряет право восстановления на приборе
Транзакт сохраняет право восстановления на приборе</td></tr>
</table>

Таблица 2.31

Операнд	Значение	Значение по умолчанию
A	Имя прибора, подлежащего возвращению	Ошибка

Операнд A блока PREEMPT задает имя прибора, подлежащего захвату. Это единственный обязательный операнд. Если все остальные операнды не используются, то захват прибора произойдет, если обслуживаемый транзакт сам не является захватчиком данного прибора.

Задание операнда B позволяет ограничить возможности транзакта-захватчика. Если в поле операнда B указано двухсимвольное выражение PR, то захват прибора произойдет только при условии, что приоритет транзакта-захватчика выше приоритета обслуживаемого транзакта.

Вопрос об использовании операнда C, по сути дела, есть вопрос о судьбе транзакта. обслуживание которого было прервано (прерванного транзакта). Если операнд C не используется, то прерванный транзакт выводится из ЦТС и помещается в отдельную область, называемую цепью прерываний данного прибора. В противном случае, прерванный транзакт перейдет в блок, имя которого задает операнд C. При этом заметим, что прерванный транзакт, помещенный в цепь прерываний, по истечении некоторого промежутка времени вернется обратно на прибор. Это произойдет при входе транзакта-захватчика в блок RETURN, единственный операнд которого задает имя прибора, подлежащего возврату (табл.2.31). После возвращения прерванного транзакта на прибор процесс обслуживания возобновляется с момента прерывания, т.е. происходит дообслуживание прерванного транзакта,

Операнд D используется в том случае, когда в один из параметров прерванного транзакта необходимо записать время, оставшееся до окончания его обслуживания. Номер параметра задается значением операнда D.

Операнд E используется лишь в том случае, если задано значение операнда C. Он указывает на возможность возвращения прерванного транзакта на дообслуживание в момент освобождения прибора. Если в поле операнда E указано двухсимволъное выражение PR, то прерванный транзакт не будет возвращен на дообслуживание. Если же операнд E не используется, то сразу после входа транзакта-захватчика в соответствующий блок RETURN прерванный транзакт будет автоматически возвращен на дообслуживание.

Ниже приведены примеры использования блоков PREEMPT и RETURN.

```
Пример 1.
PREEMPT     P2
RETURN      P2

Пример 2.
PREEMPT     DEV,PR,V8,2,RE
RETURN      DEV
```

В первом примере захват прибора с номером, равным значению второго параметра транзакта, происходит в том случае, если обслуживаемый в данный момент тракзакт сам не является захватчиком прибора. Затем при входе транзакта-захватчика в блок RETURN прибор возвращается его предыдущему владельцу.

В примере 2 захват прибора DEL происходит в том случае, если приоритет транзакта-захватчика выше, чем приоритет обслуживаемого транзакта. При этом прерванный транзакт переходит в блок с номером, равным значению восьмой переменной и после освобождения прибора, наступающего при входе транзакта в блок RETURN, дообслуживаться не будет. Время, оставшееся до окончания обслуживания прерванного транзакта, будет записано во второй параметр.

2.30. Пример моделирования 6

Постановка задачи. Построим имитационную модель однолинейной СМО, на которую поступают k независимых

пуассоновских потоков заявок. Интенсивность i-го потока обозначим λ_i , $i = 1,...,k$. Система имеет r мест для ожидания, $0 \leq r < \infty$. Если поступающая в систему заявка застает накопитель полностью занятым, то она теряется.

Времена обслуживания заявок не зависят от длины или состава очереди и от поступающего потока. Функции распределения времени обслуживания заявок i -го потока являются экспоненциальными с параметрами μ_i , $o < \mu_i < \infty$, $i=1,...,k$. Выбор заявки из очереди на обслуживание происходит в соответствии с дисциплиной FIFO.

Прибор является ненадежным. Он может выходить из строя, как во время обслуживания, так и в свободном состоянии. При этом предполагается, что распределение времени пребывания прибора в исправном состоянии является экспоненциальным со средним значением $1/\lambda_0$. Время восстановления прибора имеет экспоненциальное распределение со средним значением $1/\mu_0$. Если в момент поломки происходит обслуживание, то обслуживаемая заявка теряется.

Аналитическое исследование рассматриваемой СМО впервые проводилась в [6]. Воспользуемся рядом замечаний этой работы, относящихся к процессу функционирования СМО. Прежде всего, в соответствии с [6] будем рассматривать процесс поломок и восстановлений прибора как пуассоновский поток "вредных" заявок с интенсивностью λ_0 , обладающих свойством прерывания обслуживания заявок других потоков ("полезных" заявок). Время обслуживания "вредных" заявок распределено по экспоненциальному закону с параметром μ_0. При этом заявки "вредного" потока в очереди находиться не могут, т.е. для них в системе не существует мест для ожидания (рис. 2.2).

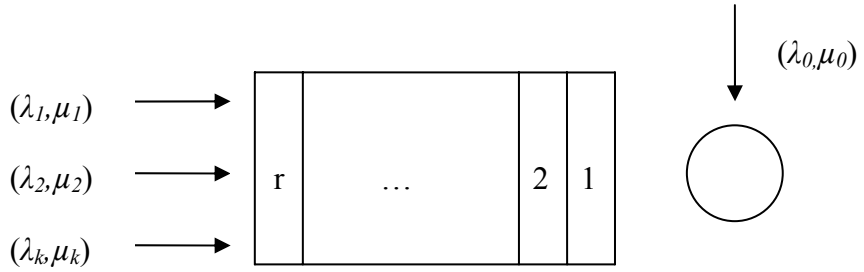

Рис. 2.2

В работах [7,8] показано, что при дисциплине FIFO выбор заявки из очереди на обслуживание не зависит от типа заявки (т.е. от номера потока, которому принадлежит данная заявка). Поэтому можно не различать заявки, находящиеся в очереди до того момента, когда начинается обслуживание, т.е. считать, что на вход поступает один пуассоновский поток с интенсивностью $\lambda = \lambda_1 + ... + \lambda_k$, а после выбора заявки из очереди для обслуживания розыгрыш ее типа производится по полиномиальной схеме с вероятностями $\alpha_1=\lambda_1/\lambda$, … $\alpha_k=\lambda_k/\lambda$.

С учетом сделанных замечаний и будем строить имитационную модель СМО.

Распечатка модели и результатов моделирования приведены в приложениях 2.6.1 и 2.6.2 соответственно.

ЛИТЕРАТУРА

1. Матюшенко С.И., Спесивов С.С. Методические рекомендации к изучению курса «Статистическое моделирование». Тема «Моделирование систем массового обслуживания на GPSS». Для студентов III-V курсов специальностей «Математика», «Прикладная математика».- М.: Изд-во УДН, 1991.- 72 с.

2. Руководство пользователя по GPSS World./Перевод с английского/.- Казань: Изд-во «Мастер Лайн», 2002.- 384 с.

3. Томашевский В.Н., Жданова Е.Г. Имитационное моделирование в среде GPSS. – М.: Бестселлер, 2003.- 416 с.

4. Шрайбер Т.Д. Моделирование на GPSS. – М.: Машиностроение, 1980. – 592 с.

5. Айвазян С.А., Енкшов И.С., Мешалкин Л.Д. Прикладная статистика: Основы моделирования и первичная обработка данных. Справочное издание. –М.: Финансы и статистика, 1983. – 471 с.

6. Башарин Г.П., Харкевич А.Д., Шнепс М.А. Массовое обслуживание в телефонии. – М.: Наука, 1968.

7. Башарин Г.П. О расчете буферной памяти в вычислительных системах с несколькими входящими информационными потоками// Системы управления и коммутации. – М.: Наука, 1965. – С.24-41.

8. Башарин Г.П. Один прибор с конечной очередью и заявками нескольких видов// Теория вероятностей и ее применения, 1965.- Т.10, вып.2. – С.282-296.

ПРИЛОЖЕНИЕ

Приложение 2.1.1. Пример моделирования 1. Распечатка модели.

```
;    Модель однолинейной СМО с накопителем бесконечной емкости и с
;    интервалами между поступлениями и длительностями обслуживания
;    заявок распределенными по равномерному закону
;
;
          GENERATE 18,6    ; поступление заявок

          QUEUE    que     ; присоединение к очереди
          SEIZE    serv    ; занятие прибора
          DEPART   que     ; уход из очереди
          ADVANCE  16,6    ; обслуживание на приборе
          RELEASE  serv    ; освобождение прибора

          TERMINATE1       ; уход из системы
```

Приложение 2.1.2. Пример моделирования 1.
Распечатка результатов.

```
                    Wednesday, January 25, 2006
20:53:18

          START TIME            END TIME  BLOCKS
FACILITIES  STORAGES
               0.000          18063.296     7
1           0

              NAME                          VALUE
           QUE                           10000.000
           SERV                          10001.000

 LABEL            LOC  BLOCK TYPE     ENTRY COUNT
CURRENT COUNT RETRY
                   1    GENERATE          1001
0       0
                   2    QUEUE             1001
0       0
                   3    SEIZE             1001
1       0
                   4    DEPART            1000
0       0
                   5    ADVANCE           1000
0       0
```

```
                  6    RELEASE           1000
0       0
                  7    TERMINATE         1000
0       0

FACILITY ENTRIES  UTIL. AVE. TIME AVAIL.OWNER
PEND INTER RETRY DELAY
 SERV     1001    0.882     15.920  1   1001
0    0     0      0


QUEUE   MAX CONT. ENTRY ENTRY(0) AVE.CONT.
AVE.TIME   AVE.(-0) RETRY
 QUE     3    1   1001    482     0.172
3.095      5.969   0
;         Приложение 2.2.1. Пример моделирования 2.
Распечатка модели.
;
;
;         Модель работы порта
;
;
prch1     STORAGE   6         ; причалы для кораблей
первого типа
prch2     STORAGE   3         ; причалы для кораблей
второго типа
buks      STORAGE   2         ; буксиры
;         корабли первого типа
          GENERATE  130,30    ; подход к порту
          QUEUE     type1     ; занять место в
очереди
          ENTER     prch1     ; получить причал
          ENTER     buks      ; получить буксир
          DEPART    type1     ; покинуть очередь
          ADVANCE   30,7      ; отбуксироваться к
причалу
          LEAVE     buks      ; освободить буксир
          ADVANCE   720,120   ; разгрузка\погрузка
          ENTER     buks      ; получить буксир
          LEAVE     prch1     ; освободить причал
```

```
          ADVANCE   20,5      ; отбуксироваться от
причала
          LEAVE     buks      ; освободить буксир
          TERMINATE           ; покинуть порт
;         корабли второго типа
          GENERATE  390,60
          QUEUE     type2
          ENTER     prch2
          ENTER     buks,2
          DEPART    type2
          ADVANCE   45,12
          LEAVE     buks,2
          ADVANCE   1080,240
          ENTER     buks,2
          LEAVE     prch2
          ADVANCE   35,10
          LEAVE     buks,2
          TERMINATE
;         наблюдать работу порта 43200 минут (30
суток)
          GENERATE  43200
          TERMINATE 1
```

Приложение 2.2.2. Пример моделирования 2. Распечатка результатов.

```
                    Thursday, January 26, 2006
14:54:08

            START TIME            END TIME  BLOCKS
FACILITIES  STORAGES
                 0.000           43200.000    28
0           3

               NAME                        VALUE
            BUKS                         10002.000
            PRCH1                        10000.000
            PRCH2                        10001.000
            TYPE1                        10003.000
            TYPE2                        10004.000

 LABEL              LOC  BLOCK TYPE     ENTRY COUNT
CURRENT COUNT RETRY
                     1    GENERATE            328
0        0
                     2    QUEUE               328
0        0
                     3    ENTER               328
0        0
                     4    ENTER               328
0        0
                     5    DEPART              328
0        0
                     6    ADVANCE             328
0        0
                     7    LEAVE               328
0        0
                     8    ADVANCE             328
6        0
                     9    ENTER               322
0        0
                    10    LEAVE               322
0        0
```

```
                    11    ADVANCE            322
0       0
                    12    LEAVE              322
0       0
                    13    TERMINATE          322
0       0
                    14    GENERATE           109
0       0
                    15    QUEUE              109
0       0
                    16    ENTER              109
1       0
                    17    ENTER              108
0       0
                    18    DEPART             108
0       0
                    19    ADVANCE            108
0       0
                    20    LEAVE              108
0       0
                    21    ADVANCE            108
2       0
                    22    ENTER              106
0       0
                    23    LEAVE              106
0       0
                    24    ADVANCE            106
1       0
                    25    LEAVE              105
0       0
                    26    TERMINATE          105
0       0
                    27    GENERATE             1
0       0
                    28    TERMINATE            1
0       0


QUEUE          MAX CONT. ENTRY ENTRY(0) AVE.CONT.
AVE.TIME   AVE.(-0) RETRY
```

```
 TYPE1          2    0    328    133     0.278
36.677     61.692   0
 TYPE2          2    1    109     19     0.414
164.080    198.719   0


STORAGE         CAP. REM. MIN. MAX.  ENTRIES AVL.
AVE.C. UTIL. RETRY DELAY
 PRCH1           6    0   0     6      328   1
5.713  0.952    0    0
 PRCH2           3    0   0     3      109   1
2.841  0.947    0    0
 BUKS            2    0   0     2     1078   1
0.770  0.385    0    1
;         Приложение 2.3.1. Пример моделирования 3.
Распечатка модели.
;
;
;         Модель однолинейной СМО конечной емкости с
двумя типами заявок,
;         пуассоновским потоком, детерминированным
обслуживанием и от-
;         носительным приоритетом
;
que       STORAGE   5                    ; задание
емкости накопителя
;         сегмент 1 - генерация заявок первого типа
          GENERATE(exponential(1,0,18)) ; генерация
заявок первого типа
          PRIORITY  1                    ; присвоение
приоритета
          ASSIGN    1,12                 ; занесение
в первый параметр
;                                          заявки ее
длительности обслуж.
          TRANSFER  ,seg3                ; передача
управления 3-му сег-
;                                          менту
;         сегмент 2 - генерация заявок второго типа
          GENERATE(exponential(1,0,15)) ; генерация
заявок второго типа
```

```
          ASSIGN    1,8                 ; занесение
в первый параметр
;                                       заявки ее
длительности обслуж.
;         сегмент 3 - приход заявок в систему и
обслуживание
seg3      TRANSFER  both,,bye           ; потери
заявок при занятом
;                                       накопителе
          ENTER     que                 ;
присоединение к очереди
          SEIZE     serv                ; занятие
прибора
          LEAVE     que                 ; уход из
очереди
          ADVANCE   P1                  ;
обслуживание на приборе
          RELEASE   serv                ;
освобождение прибора
          TERMINATE                     ; уход
обслуженных заявок
bye       TERMINATE                     ; уход
потерянных заявок
;         сегмент 4 - задание момента окончания
прогона модели
          GENERATE  500000
          TERMINATE 1
```

Приложение 2.3.2. Пример моделирования 3. Распечатка результатов.

```
                    GPSS World Simulation Report - ex3.1.1


                    Thursday, January 26, 2006 15:18:19

            START TIME            END TIME  BLOCKS FACILITIES  STORAGES
                 0.000          500000.000    16          1          1


               NAME                          VALUE
          BYE                               14.000
          QUE                            10000.000
          SEG3                               7.000
          SERV                           10001.000


LABEL              LOC  BLOCK TYPE     ENTRY COUNT CURRENT COUNT RETRY
                     1    GENERATE          27768         0       0
                     2    PRIORITY          27768         0       0
                     3    ASSIGN            27768         0       0
```

```
                    4    TRANSFER         27768
0       0
                    5    GENERATE         33126
0       0
                    6    ASSIGN           33126
0       0
SEG3                7    TRANSFER         60894
0       0
                    8    ENTER            49144
3       0
                    9    SEIZE            49141
0       0
                   10    LEAVE            49141
0       0
                   11    ADVANCE          49141
1       0
                   12    RELEASE          49140
0       0
                   13    TERMINATE        49140
0       0
BYE                14    TERMINATE        11750
0       0
                   15    GENERATE             1
0       0
                   16    TERMINATE            1
0       0


FACILITY     ENTRIES  UTIL.   AVE. TIME AVAIL. OWNER
PEND INTER RETRY DELAY
 SERV        49141    0.966       9.825  1    60893
0    0     0      3


STORAGE          CAP. REM. MIN. MAX.  ENTRIES AVL.
AVE.C. UTIL. RETRY DELAY
 QUE              5    2   0     5    49144   1
3.022  0.604    0    0

;         Приложение 2.4.1. Пример моделирования 4.
Распечатка модели.
```

```
;
;
;          Модель однолинейной СМО конечной емкости с
двумя типами заявок:
;          основными и фоновыми. Интервалы времени
между поступлениями
;          основных заявок и длительности их
обслуживания имеют гипер-
;          экспоненциальное распределение. Фоновые
заявки поступают в
;          систему немедленно после освобождения
системы. Длительности их
;          обслуживания имеют гиперэкспоненциальное
распределение отличное
;          от распределения длительностей
обслуживания основных заявок.
;          Ни основные, ни фоновые заявки не
прерывают обслуживания друг
;          друга.
;
           INITIAL   X$NAK,5            ; задание
емкости накопителя
ALFA       FUNCTION  RN1,D3             ; задание
параметров гиперэкс-
0.2,1/0.7,2/1,3                         ;
поненциального распределения
ALAM       MATRIX    ,1,3               ; интервалов
между поступлениями
           INITIAL   MX$ALAM(1,1),2     ; основных
заявок
           INITIAL   MX$ALAM(1,2),3
           INITIAL   MX$ALAM(1,3),1
ABETA      FUNCTION  RN2,D3             ; задание
параметров гиперэкс-
0.3,1/0.7,2/1,3                         ;
поненциального распределения
AMU        MATRIX    ,1,3               ;
длительностей обслуживания
           INITIAL   MX$AMU(1,1),1      ; основных
заявок
           INITIAL   MX$AMU(1,2),2
```

```
          INITIAL   MX$AMU(1,3),3
AGAMMA    FUNCTION  RN3,D2             ; задание
параметров гиперэкс-
0.4,1/1,2                              ;
поненциального распределения
SIGMA     MATRIX    ,1,2               ;
длительностей обслуживания
          INITIAL   MX$SIGMA(1,1),1    ; фоновых
заявок
          INITIAL   MX$SIGMA(1,2),2
TIME1     TABLE     MP5,0,2,50         ; задание
таблицы для определе-
;                                        ния времени
пребывания основ-
;                                        ных заявок в
системе
TIME2     TABLE     MP6,0,2,50         ; задание
таблицы для определе-
;                                        ния времени
пребывания фоно-
;                                        вых заявок в
системе
;
          GENERATE  ,,,1               ; генерация
первой основной заявки
AAA3      ASSIGN    1,FN$ALFA          ; розыгрыш
фазы генерации
;
          ADVANCE(exponential(4,0,MX$ALAM(1,P1))) ;
задержка на фазе
;                                        генерации
          SPLIT     1,AAA3             ; генерация
следующей основной
;                                        заявки
          MARK      5                  ; отметка
времени поступления
;                                        в СМО
основной заявки
          TEST L    Q$QUE,X$NAK,AAA1   ; проверка
наличия места в
;                                        накопителе
```

```
          QUEUE     QUE               ; поступление
в очередь
          SEIZE     DEV               ; занятие
прибора основной
;                                     заявкой
          DEPART    QUE               ; уход из
очереди
          ASSIGN    2,FN$ABETA        ; розыгрыш
фазы обслуживания
;                                     основной
заявки
          ADVANCE(exponential(5,0,MX$AMU(1,P2))) ;
обслуживание ос-
;                                     новной
заявки
          TABULATE  TIME1             ; табуляция
времени пребывания
;                                     в СМО
основной заявки
AAA6      TEST E    Q$QUE,0,AAA2      ; будет
система пуста после ухода
;                                     очередной
заявки (основной либо
;                                     фоновой)
          MARK      6                 ; отметка
времени поступления
;                                     в СМО
фоновой заявки
          ASSIGN    3,FN$AGAMMA       ; розыгрыш
фазы обслуживания
;                                     фоновой
заявки

          ADVANCE(exponential(6,0,MX$SIGMA(1,P3))) ;
обслуживание фоно-
;                                     вой заявки

          TABULATE  TIME2             ; табуляция
времени пребывания
;                                     в СМО
фоновой заявки
```

```
          TRANSFER  ,AAA6              ; перейти к
проверке: будет ли
;                                      система
пуста после ухода заявки?
AAA2      RELEASE   DEV               ; освобождение
прибора
          TERMINATE                   ; уход
обслуженной заявки (основной
;                                      либо
фоновой)
AAA1      TERMINATE                   ; уход
потерянной основной заявки
          GENERATE  500000            ; генерация
таймера
          TERMINATE 1
```

Приложение 2.4.2. Пример моделирования 4. Распечатка результатов.

```
                Thursday, January 26, 2006
17:55:19

           START TIME           END TIME  BLOCKS
FACILITIES  STORAGES
                0.000         500000.000    23
1          0

              NAME                        VALUE
          AAA1                            21.000
          AAA2                            19.000
          AAA3                             2.000
          AAA6                            13.000
          ABETA                        10003.000
```

AGAMMA	10005.000
ALAM	10002.000
ALFA	10001.000
AMU	10004.000
DEV	10010.000
NAK	10000.000
QUE	10009.000
SIGMA	10006.000
TIME1	10007.000
TIME2	10008.000

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
	1	GENERATE	1	0	0
AAA3	2	ASSIGN	227458	0	0
	3	ADVANCE	227458	1	0
	4	SPLIT	227457	0	0
	5	MARK	227457	0	0
	6	TEST	227457	0	0
	7	QUEUE	191465	5	0
	8	SEIZE	191460	0	0
	9	DEPART	191460	0	0
	10	ASSIGN	191460	0	0
	11	ADVANCE	191460	1	0
	12	TABULATE	191459	0	0
AAA6	13	TEST	264186	0	0

```
                    14    MARK              72727
0       0
                    15    ASSIGN            72727
0       0
                    16    ADVANCE           72727
0       0
                    17    TABULATE          72727
0       0
                    18    TRANSFER          72727
0       0
AAA2                19    RELEASE          191459
0       0
                    20    TERMINATE        191459
0       0
AAA1                21    TERMINATE         35992
0       0
                    22    GENERATE              1
0       0
                    23    TERMINATE             1
0       0

FACILITY     ENTRIES  UTIL.   AVE. TIME AVAIL. OWNER
PEND INTER RETRY DELAY
 DEV         191460    1.000       2.611  1   227452
0    0     0      5


QUEUE           MAX CONT. ENTRY ENTRY(0) AVE.CONT.
AVE.TIME   AVE.(-0) RETRY
 QUE              5    5 191465      1     2.184
5.705      5.705   0

TABLE           MEAN    STD.DEV.       RANGE
RETRY FREQUENCY CUM.%
 TIME1          7.708    5.388
0
                                0.000  -        2.000
20145    10.52
                                2.000  -        4.000
33858    28.21
```

33239	45.57	4.000	-	6.000
28428	60.42	6.000	-	8.000
22745	72.29	8.000	-	10.000
17114	81.23	10.000	-	12.000
12156	87.58	12.000	-	14.000
8439	91.99	14.000	-	16.000
5633	94.93	16.000	-	18.000
3584	96.80	18.000	-	20.000
2328	98.02	20.000	-	22.000
1449	98.78	22.000	-	24.000
925	99.26	24.000	-	26.000
577	99.56	26.000	-	28.000
355	99.75	28.000	-	30.000
200	99.85	30.000	-	32.000
113	99.91	32.000	-	34.000
61	99.94	34.000	-	36.000
45	99.97	36.000	-	38.000
26	99.98	38.000	-	40.000
17	99.99	40.000	-	42.000
13	100.00	42.000	-	44.000

```
                                    44.000  -       46.000
4   100.00
                                    46.000  -       48.000
2   100.00
                                    48.000  -       50.000
0   100.00
                                    50.000  -       52.000
3   100.00


TIME2           1.602    1.741
0
                                     0.000  -        2.000
52763    72.55
                                     2.000  -        4.000
13511    91.13
                                     4.000  -        6.000
4215    96.92
                                     6.000  -        8.000
1405    98.85
                                     8.000  -       10.000
543    99.60
                                    10.000  -       12.000
196    99.87
                                    12.000  -       14.000
57    99.95
                                    14.000  -       16.000
24    99.98
                                    16.000  -       18.000
7    99.99
                                    18.000  -       20.000
6   100.00


SAVEVALUE                RETRY       VALUE
 NAK                       0          5.000


;         Приложение 2.5.1. Пример моделирования 5.
Распечатка модели.
;
```

```
;
;         Модель однопотоковой многоканальной СМО
конечной емкости с
;         разделяющимися заявками и буфером
синхронизации
;
;         Задание исходных данных
          INITIAL   X$KDEV,3              ; задание
количества приборов
          INITIAL   X$NAK,3               ; задание
емкости накопителя
          INITIAL   X$ALAM,10             ; задание
средней длины интервала
;                                           между
поступлениями заявок
          INITIAL   X$NLOST,0             ;
обнуление счетчика потерянных
;                                           заявок
          INITIAL   X$FLAG,0              ;
обнуление индикатора занятости
;                                          приборов:
0 -свободны, 1- заняты
          INITIAL   X$NSERV,0             ;
обнуление счетчика числа выпол-
;                                           ненных
заданий
PROB      MATRIX    ,1,3                  ; задание
матрицы вероятностей
;
присутствия соотв. задания в
;                                           заявке
INTEN     MATRIX    ,1,3                  ; задание
матрицы средних дли-
;
тельностей обслуживания на при-
;                                           борах
;                                           задание
элементов соответству-
          INITIAL   MX$PROB(1,1),800      ; ющих
матриц
          INITIAL   MX$PROB(1,2),900
```

```
          INITIAL   MX$PROB(1,3),950
          INITIAL   MX$INTEN(1,1),5
          INITIAL   MX$INTEN(1,2),4
         INITIAL   MX$INTEN(1,3),4
RESPT    TABLE     MP2,0,4,50            ; задание
таблицы, определяющей
;                                          длительность
пребывания заявок
;                                          в СМО
         GENERATE(exponential(1,0,X$ALAM)); генерация
входящего потока заявок
         TEST L    Q$QUE,X$NAK,AAA5      ; проверка
заполненности накопителя
          MARK      2                     ; отметка
времени поступления заявки
          QUEUE     QUE                   ;
поступление заявки в очередь
          TEST E    X$FLAG,0              ;
проверка: свободны ли приборы?
          DEPART    QUE                   ; уход
заявки из очереди
          SAVEVALUE FLAG,1                ;
установить индикатор в положение:
;                                           приборы
заняты
          SPLIT     (X$KDEV-1),AAA1,1     ;
разбиение заявки на задания
AAA1      TEST LE   RN2,MX$PROB(1,P1),AAA2;
просеивание заданий
          SEIZE     P1                    ; занятие
прибора заданием
          ADVANCE(exponential(3,0,MX$INTEN(1,P1)));
выполнение задания
          RELEASE   P1                    ;
освобождение прибора
AAA2      SAVEVALUE NSERV+,1              ;
увеличение счетчика числа выпол-
;                                          ненных
заданий
          TEST L    X$NSERV,X$KDEV,AAA3   ;
проверка: задание является пос-
```

```
;                                                  ледним?
          LINK      BUFF,P1                 ;
поместить задание в буфер син-
;
хронизации
AAA3      SAVEVALUE FLAG,0                  ;
установить индикатор в положение:
;                                                  приборы
свободны
          SAVEVALUE NSERV,0                 ;
обнуление счетчика числа выполнен-
;                                                  ных
заданий
          TABULATE  RESPT                   ;
табулирование времени пребывания
;                                                  заявок в
СМО
          UNLINK    BUFF,AAA4,ALL           ; вывод
заданий из буфера синхрони-
;                                                  зации
AAA4      TERMINATE 1                       ; уход
заданий из системы
AAA5      SAVEVALUE NLOST+,1                ;
увеличение счетчика потерянных
;                                                  заявок
          TERMINATE
```

Приложение 2.5.2. Пример моделирования 5. Распечатка результатов.

```
                    GPSS World Simulation Report - ex5.2.1

                    Thursday, January 26, 2006 19:52:02

            START TIME            END TIME  BLOCKS FACILITIES  STORAGES
                0.000           54416.070    22          3          0


               NAME                          VALUE
          AAA1                                9.000
          AAA2                               13.000
          AAA3                               16.000
          AAA4                               20.000
          AAA5                               21.000
          ALAM                            10002.000
          BUFF                            10010.000
          FLAG                            10004.000
          INTEN                           10007.000
          KDEV                            10000.000
          NAK                             10001.000
          NLOST                           10003.000
          NSERV                           10005.000
          PROB                            10006.000
          QUE                             10009.000
          RESPT                           10008.000
```

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
	1	GENERATE	5395	0	0
	2	TEST	5395	0	0
	3	MARK	5003	0	0
	4	QUEUE	5003	2	0
	5	TEST	5001	0	0
	6	DEPART	5001	0	0
	7	SAVEVALUE	5001	0	0
	8	SPLIT	5001	0	0
AAA1	9	TEST	15001	0	0
	10	SEIZE	13202	0	0
	11	ADVANCE	13202	1	0
	12	RELEASE	13201	0	0
AAA2	13	SAVEVALUE	15000	0	0
	14	TEST	15000	0	0
	15	LINK	10000	0	0
AAA3	16	SAVEVALUE	5000	0	0
	17	SAVEVALUE	5000	0	0
	18	TABULATE	5000	0	0
	19	UNLINK	5000	0	0

```
AAA4                20    TERMINATE        15000
0       0
AAA5                21    SAVEVALUE          392
0       0
                    22    TERMINATE          392
0       0


FACILITY      ENTRIES  UTIL.   AVE. TIME AVAIL. OWNER
PEND INTER RETRY DELAY
 1              3963    0.365       5.014  1    15389
0    0     0       0
 2              4489    0.337       4.082  1        0
0    0     0       0
 3              4750    0.354       4.051  1        0
0    0     0       0


QUEUE            MAX CONT. ENTRY ENTRY(0) AVE.CONT.
AVE.TIME   AVE.(-0) RETRY
 QUE               3    2   5003   1672     0.698
7.588     11.397   0


TABLE              MEAN    STD.DEV.       RANGE
RETRY FREQUENCY CUM.%
 RESPT            14.982   10.048
0
                                                 -
0.000              1     0.02                  _
                                          0.000  -
4.000            539    10.80
                                          4.000  -
8.000            948    29.76
                                          8.000  -
12.000           799    45.74
```

```
                                        12.000  -
16.000           753    60.80
                                        16.000  -
20.000           555    71.90
                                        20.000  -
24.000           481    81.52
                                        24.000  -
28.000           363    88.78
                                        28.000  -
32.000           234    93.46
                                        32.000  -
36.000           138    96.22
                                        36.000  -
40.000            82    97.86
                                        40.000  -
44.000            51    98.88
                                        44.000  -
48.000            36    99.60
                                        48.000  -
52.000            12    99.84
                                        52.000  -
56.000             5    99.94
                                        56.000  -
60.000             0    99.94
                                        60.000  -
64.000             1    99.96
                                        64.000  -
68.000             1    99.98
                                        68.000  -
72.000             1   100.00


USER CHAIN          SIZE RETRY  AVE.CONT   ENTRIES
MAX     AVE.TIME
 BUFF                  0    0     0.983    10000
2        5.350

SAVEVALUE               RETRY       VALUE
 KDEV                     0          3.000
 NAK                      0          3.000
 ALAM                     0         10.000
```

```
 NLOST                    0        392.000
 FLAG                     2          1.000
 NSERV                    0              0

CEC XN   PRI          M1      ASSEM  CURRENT  NEXT
PARAMETER    VALUE
 15392    0       54405.797   15392      4      5
2      54405.797
 15395    0       54408.368   15395      4      5
2      54408.368
 15397    0       54403.855   15389      0      9
1          2.000

2      54403.855
 15398    0       54403.855   15389      0      9
1          3.000

2      54403.855

FEC XN   PRI          BDT      ASSEM  CURRENT  NEXT
PARAMETER    VALUE
 15396    0       54429.366   15396      0      1
 15389    0       54432.895   15389     11     12
1          1.000

2      54403.855

;         Приложение 2.6.1. Пример моделирования 6.
Распечатка модели.
;
;         Модель однолинейной СМО конечной емкости с
k независимыми пуассо-
;         новскими потоками заявок, экспоненциальным
обслуживанием и нена-
;         дежным прибором
;
;
          INITIAL   X$CAP,6                ; задание
емкости накопителя
          INITIAL   X$TYP,5                ; задание
количества типов заявок
```

```
          INITIAL   X$WORK,200             ; задание
времени пребывания при-
;                                           бора в
исправном состоянии
          INITIAL   X$ARV,25               ; задание
ср. времени между пос-
;
туплениями заявок
          INITIAL   X$REP,1                ; задание
ср. времени восстанов-
;                                           ления
прибора
SERV      MATRIX    ,1,5                   ; задание
матрицы длительностей
;
обслуживания заданий
          INITIAL   MX$SERV(1,1),25
          INITIAL   MX$SERV(1,2),17
          INITIAL   MX$SERV(1,3),12
          INITIAL   MX$SERV(1,4),10
          INITIAL   MX$SERV(1,5),8
          INITIAL   X$LOST,0               ;
обнуление счетчика числа поте-
;                                           рянных
заданий
          INITIAL   X$BREAK,0              ;
обнуление числа поломок
          INITIAL   X$CAT,0                ;
обнуление числа прерываний
PROB      FUNCTION  RN1,D5                 ; задание
ФР заданий по типам
.2,1/.35,2/.5,3/.7,4/1,5
          GENERATE(exponential(1,0,X$ARV)) ;
генерация полезных заявок
          ASSIGN    1,FN$PROB              ; запись
типа заявки в P1
          ASSIGN    2,1                    ; запись
признака полезности в P2
          TEST L    Q$QUE,X$CAP,BYE2       ;
проверка занятости накопителя
```

```
          QUEUE     QUE                     ;
поступление в очередь
          SEIZE     DEV                     ; занятие
прибора
          DEPART    QUE                     ; уход из
очереди
          ADVANCE(exponential(2,0,MX$SERV(1,P1))) ;
обслуживание заявки
          RELEASE   DEV                     ;
освобождение прибора
          TERMINATE 1                       ; уход
обслуженной заявки
          GENERATE(exponential(3,0,X$WORK));
генерация вредных заявок
          ASSIGN    2,0                     ; запись
признака вредности в P2
          PREEMPT   DEV,,BYE1,,RE           ; поломка
прибора
          ADVANCE(exponential(3,0,X$REP))   ;
восстановление прибора
          RETURN    DEV                     ; возврат
прибора к работе
          SAVEVALUE BREAK+,1                ; подсчет
числа поломок
          TERMINATE                         ; уход
вредной заявки
BYE1      TEST E    P2,1,BYE3               ;
обслуживание вредной или по-
;                                             лезной
заявки было прервано?
          SAVEVALUE CUT+,1                  ; подсчет
числа прерываний
          TERMINATE                         ; уход
недообслуженной полезной
;                                             заявки
BYE2      SAVEVALUE LOST+,1                 ; подсчет
потерянных при поступ-
;                                             лении
заявок
          TERMINATE                         ; уход
потерянной на входе заявки
```

```
BYE3      TERMINATE                          ; уход
недообслуженных вредных
;                                              заявок
```

Приложение 2.6.2. Пример моделирования 6. Распечатка результатов.

```
           START TIME           END TIME  BLOCKS
FACILITIES  STORAGES
                0.000        2698508.357    23
1           0
```

NAME	VALUE
ARV	10003.000
BREAK	10007.000
BYE1	18.000
BYE2	21.000
BYE3	23.000
CAP	10000.000
CAT	10008.000
CUT	10012.000
DEV	10011.000
LOST	10006.000
PROB	10009.000
QUE	10010.000
REP	10004.000
SERV	10005.000
TYP	10001.000
WORK	10002.000

```
 LABEL              LOC  BLOCK TYPE     ENTRY COUNT
CURRENT COUNT RETRY
                   1    GENERATE         107724
0       0
                   2    ASSIGN           107724
0       0
                   3    ASSIGN           107724
0       0
                   4    TEST             107724
0       0
```

		5	QUEUE	106837	
0	0				
		6	SEIZE	106837	
0	0				
		7	DEPART	106837	
0	0				
		8	ADVANCE	106837	
0	0				
		9	RELEASE	100000	
0	0				
		10	TERMINATE	100000	
0	0				
		11	GENERATE	13438	
0	0				
		12	ASSIGN	13438	
0	0				
		13	PREEMPT	13438	
0	0				
		14	ADVANCE	13438	
0	0				
		15	RETURN	13358	
0	0				
		16	SAVEVALUE	13358	
0	0				
		17	TERMINATE	13358	
0	0				
BYE1		18	TEST	6917	
0	0				
		19	SAVEVALUE	6837	
0	0				
		20	TERMINATE	6837	
0	0				
BYE2		21	SAVEVALUE	887	
0	0				
		22	TERMINATE	887	
0	0				
BYE3		23	TERMINATE	80	
0	0				

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.		
OWNER	PEND	INTER	RETRY	DELAY		
DEV	120275	0.508	11.397	1	0	0
0	0	0				

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	
AVE.TIME	AVE.(-0)	RETRY				
QUE	6	0	106837	52945	0.558	
14.087	27.926	0				

SAVEVALUE	RETRY	VALUE
CAP	0	6.000
TYP	0	5.000
WORK	0	200.000
ARV	0	25.000
REP	0	1.000
LOST	0	887.000
BREAK	0	13358.000
CAT	0	0
CUT	0	6837.000

СОДЕРЖАНИЕ

ВВЕДЕНИЕ…….3
ГЛАВА 1. ОСНОВЫ СТАТИСТИЧЕСКОГО МОДЕЛИРОВАНИЯ……..5
1.1. Метод статистического моделирования…….....5
1.2. Моделирование случайных событий и дискретных случайных величин……...6
1.3. Моделирование непрерывных случайных величин…....8
ГЛАВА 2. ОСНОВНЫЕ ЭЛЕМЕНТЫ GPSS……..……...15
2.1. Общая структура модели на языке GPSS……...15
2.2. Таймер модельного времени…….……17
2.3. Общая структура блоков и операторов……17
2.4. Блок GENERATE……...……..…19
2.5. Блок TERMINATE, команда START…….....21
2.6. Блоки SEIZE и RELEASE……...22
2.7. Блок ADVANCE……....23
2.8. Блоки QUEUE и DEPART……....24
2.9. Пример моделирования 1……...25
2.9.1. Постановка задачи…….....25
2.9.2. Логика моделирования……25
2.9.3. Результаты моделирования……..26
2.10. Цепи текущих и будущих событий. Логика работы интерпретатора…… ……...28
2.11. Распределения вероятностей в GPSS…… ……....34

2.11.1. Генераторы равномерно распределенных случайных чисел……..…..34
2.11.2. Получение равномерных распределений…….....34
2.11.3. Определение дискретных функций в GPSS…...…34
2.11.4. Использование дискретных функций в блоках GENERATE и ADVANCE……....36.
2.11.5. Определение непрерывных функций в GPSS…...36
2.11.6. Моделирование вероятностных функций распределения в GPSS World……..…....37
2.12. Блоки ENTER, LEAVE. Оператор STORAGE…….41
2.13. Пример моделирования 2…….....44
2.14. Блок TRANSFER……...….45
2.14.1. Блок TRANSFER в режиме безусловной передачи…..……..…45
2.14.2. Блок TRANSFER в режиме условной передачи……46
2.14.3. Блок TRANSFER в режиме статистической передачи……47
2.15. Стандартные числовые атрибуты……..48
2.16. Параметры транзактов. Блок ASSIGN……..50
2.17. Блок PRIORITY……53
2.18. Пример моделирования 3……54
2.19. Переменные…….....54
2.19.1. Арифметические переменные…….54
2.19.2. Арифметические переменные с плавающей точкой……56
2.19.3. Булевы переменные…….…57
2.20. Оператор INITIAL и блок SAVEVALUE……58
2.21. Оператор MATRIX и блок MSAVEVALUE…….60
2.22. Блок TEST……....63
2.23. Блок SPLIT……...…64
2.24. Оператор TABLE и блок TABULATE……..66
2.25. Блок MARK……..…68
2.26. Пример моделирования 4……...69
2.27. Блоки LINK и UNLINK……...70
2.28. Пример моделирования 5……..……72
2.29. Блоки PREEMPT и RETURN……..…74

2.30. Пример моделирования 6……..77
ЛИТЕРАТУРА……80
ПРИЛОЖЕНИЕ……..81
2.1.1. Пример моделирования 1. Распечатка модели…….82
2.1.2. Пример моделирования 1. Распечатка результатов…….83
2.2.1. Пример моделирования 2. Распечатка модели…….84
2.2.2. Пример моделирования 2. Распечатка результатов…….86
2.3.1. Пример моделирования 3. Распечатка модели…….88
2.3.2. Пример моделирования 3. Распечатка результатов…….90
2.4.1. Пример моделирования 4. Распечатка модели…….92
2.4.2. Пример моделирования 4. Распечатка результатов…….95
2.5.1. Пример моделирования 5. Распечатка модели…….99
2.5.2. Пример моделирования 5. Распечатка результатов…..…102
2.6.1. Пример моделирования 6. Распечатка модели……...106
2.6.2. Пример моделирования 6. Распечатка результатов……108