Министерство образования и науки Российской Федерации
Государственное образовательное учреждение
высшего профессионального образования
«Кузбасский государственный технический университет»

В. В. Зиновьев, А. Н. Стародубов

МОДЕЛИРОВАНИЕ СИСТЕМ ПРИ ПОМОЩИ КОМПЬЮТЕРНОЙ ИМИТАЦИИ И АНИМАЦИИ

Учебное пособие

Кемерово 2010

УДК 519.876.5

Рецензенты:

Профессор кафедры Автоматики Новосибирского государственного технического университета, доктор технических наук В.Л. Конюх.

Заведующий кафедрой Автоматизации производственных процессов и автоматизированных систем управления Кемеровского технологического института пищевой промышленности, кандидат технических наук А.В. Чупин.

Зиновьев, В. В. Моделирование систем при помощи компьютерной имитации и анимации : учеб. пособие / В. В. Зиновьев, А. Н. Стародубов; Кузбас. гос. техн. ун-т. – Кемерово, 2010. – 118 с.

ISBN 978-5-89070-757-4

Приводятся основные понятия, необходимые для моделирования производственных процессов в дискретных технологических системах методами компьютерной имитации и анимации с использованием пакетов лицензионного программного обеспечения GPSS/H и Proof Animation (ф. Wolverine Soft. Corp.), примеры компьютерной имитации и анимации, позволяющие лучше усвоить особенности языков. В приложениях даны листинги программ примеров и файлов отчета с результатами моделирования.

Предназначено для студентов специальности 220301 «Автоматизация технологических процессов (в машиностроении)».

Печатается по решению редакционно-издательского совета Кузбасского государственного технического университета.

УДК 519.876.5

ISBN 978-5-89070-757-4

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ

В настоящее время имитационное моделирование стало эффективным средством решения задач автоматизации исследований технологических процессов в машиностроении. Добиться значительного упрощения и ускорения процесса разработки имитационных моделей возможно за счет использования специализированных языков и систем компьютерной имитации и анимации (ARENA, EXTEND, GPSS, SIMAN, SLAM, SIMULA, GASP, Proof Animation и др.), позволяющих отображать динамику работы оборудования на мнемосхеме технологического процесса в ходе имитационного моделирования.

Дисциплина «Компьютерная имитация и анимация» основана на лицензионном программном обеспечении – специализированных языках компьютерной имитации GPSS/H и анимации Proof Animation (ф. Wolverine Software Corp., USA). В настоящее время комплекс этих языков является одним из наиболее эффективных и распространенных программных средства моделирования сложных дискретных систем на ЭВМ. Он успешно используется для моделирования систем, формализуемых в виде схем массового обслуживания, с помощью которых описываются многие объекты, рассматриваемые при подготовке студентов специальности 220301 «Автоматизация технологических процессов (в машиностроении)».

В курсе «Компьютерная имитация и анимация» студенты должны изучить основные понятия имитационного моделирования, отличия, преимущества и недостатки универсальных и специализированных языков компьютерной имитации и анимации, этапы имитационного моделирования технологических процессов, принципы построения и программной реализации моделирующих алгоритмов, методы анализа и интерпретации результатов моделирования, способы и средства графического отображения статических и динамических элементов технологий в Proof Animation, методы управления анимацией.

По окончании обучения специалисты смогут разрабатывать модели технологических процессов, используя специализированный язык компьютерной имитации GPSS/H, создавать в соответствии с имитационной моделью движущееся отображение технологического процесса на экране компьютера, используя язык Proof Ani-

mation, проводить имитационные эксперименты с моделями технологических процессов, оценивать длительность производственного цикла и коэффициенты использования оборудования, принимать решения по сокращению времени простоя оборудования, сравнивать варианты организации технологического процесса и выбирать наиболее оптимальный вариант, прогнозировать поведение системы в ускоренном времени.

Разработанный учебный курс апробирован в течение пяти лет при обучении студентов 3-4 курса специальностей «Автоматизация технологических процессов (в машиностроении)» и «Информационные системы и технологии» механико-машиностроительного факультета Кузбасского государственного технического университета.

Данное учебное пособие предлагается как основное приложение к дисциплине «Компьютерная имитация и анимация» и знакомит студентов с практическими возможностями использования современных методов имитационного моделирования при проектировании и анализе технологических систем, в качестве формальных моделей которых используют системы массового обслуживания.

ВВЕДЕНИЕ

Если технологическая система не поддается физическому эксперименту, прибегают к математическому моделированию, т.е. отражению некоторых свойств системы совокупностью математических объектов (чисел, переменных, матриц, множеств, точек, отрезков прямых и т.д.) и отношений между ними. Математическое моделирование систем делят на аналитическое и имитационное. Аналитическое моделирование представляет собой функциональные соотношения (алгебраические, интегро-дифференциальные, конечно-разностные) или логические условия. Во многих случаях невозможно получить аналитические зависимости, отображающие поведение и взаимосвязь элементов технологической системы. Особенно трудно учесть действие случайных факторов и динамику функционирования объекта. Поэтому используют имитационное моделирование. Имитационное моделирование похоже на физические эксперименты, но эти эксперименты проводятся не на физическом объекте, а на его компьютерной модели. Сущность имитационного моделирования состоит в том, что над моделью проводят эксперименты типа «Что, если…?». Изменяя исходные показатели в модели, анализируют результаты экспериментов. Имитационная модель реализуется на ЭВМ при помощи языков программирования. Очень часто имитационные модели сложных технологических систем разрабатывают с использованием универсальных языков, таких как Паскаль, C^{++}, Delphi. Функционирование системы описывают последовательностью уравнений. Затем кодируют их на языке программирования и вводят программу в компьютер. Потом производят расчеты, изменяя некоторые коэффициенты. Разработка таких программ требует нескольких человеко-месяцев труда специалистов по технологии, математике и программированию. Модель получается достаточно громоздкой, в результате затрудняется ее исправление и дополнение. Зачастую разработка модели отстает от развития производства, и моделирование теряет смысл.

Альтернативой является использование методов, основанных на специализированных языках и системах имитационного моделирования ARENA, GPSS, SIMAN, SLAM, SIMSCRIPT, SIMULA, GASP и др. Специализированные языки содержат, как правило, написанные на универсальном языке блоки – отдельные динамические

модели. Поведение системы имитируется как смена ее состояний. Процесс отображается не системой уравнений, а взаимодействием отдельных динамических моделей во времени и пространстве. Поведение системы описывается от события к событию, означающих начало или окончание технологической операции.

Одним из наиболее распространенных специализированных языков имитационного моделирования является GPSS, предложенный фирмой IBM в 1962 г. Он развивался во многих версиях. В настоящем учебном пособии рассмотрена предпоследняя версия языка – GPSS/H.

Недостатком имитационного моделирования является то, что необходимо интерпретировать результаты моделирования. Сам процесс имитации скрыт от экспериментатора. Поэтому специализированные языки имитационного моделирования соединяются с программами компьютерной анимации (CIMAN – CINEMA, GPSS/H – Proof Animation), позволяющими отображать динамику процесса на экране компьютера в соответствии с работой имитационной модели и выводить результаты на монитор.

1. КЛАССИФИКАЦИЯ МОДЕЛЕЙ

Модель – описание системы. Во многих изданиях посвященных моделированию, приводится классификация методов моделирования. Одна из наиболее полных классификаций приведена в учебнике [1]. Мы рассмотрим упрощенную, но позволяющую понять, какой области моделирования посвящена дисциплина «Компьютерная имитация и анимация», классификацию. При этом остановимся на некоторых основных понятиях теории моделирования.

Все моделирование технологических систем можно разделить на три вида: реальное, физическое и математическое (рис. 1).

При физическом моделировании модель по физической природе и геометрическим формам подобна объекту моделирования, но изучается в другом масштабе пространства и времени. Характеристики объекта получают путем пересчета характеристик модели по масштабным коэффициентам. При математическом моделировании объект описывают в виде математических зависимостей, отображающих связь между входными и выходными параметрами объекта.

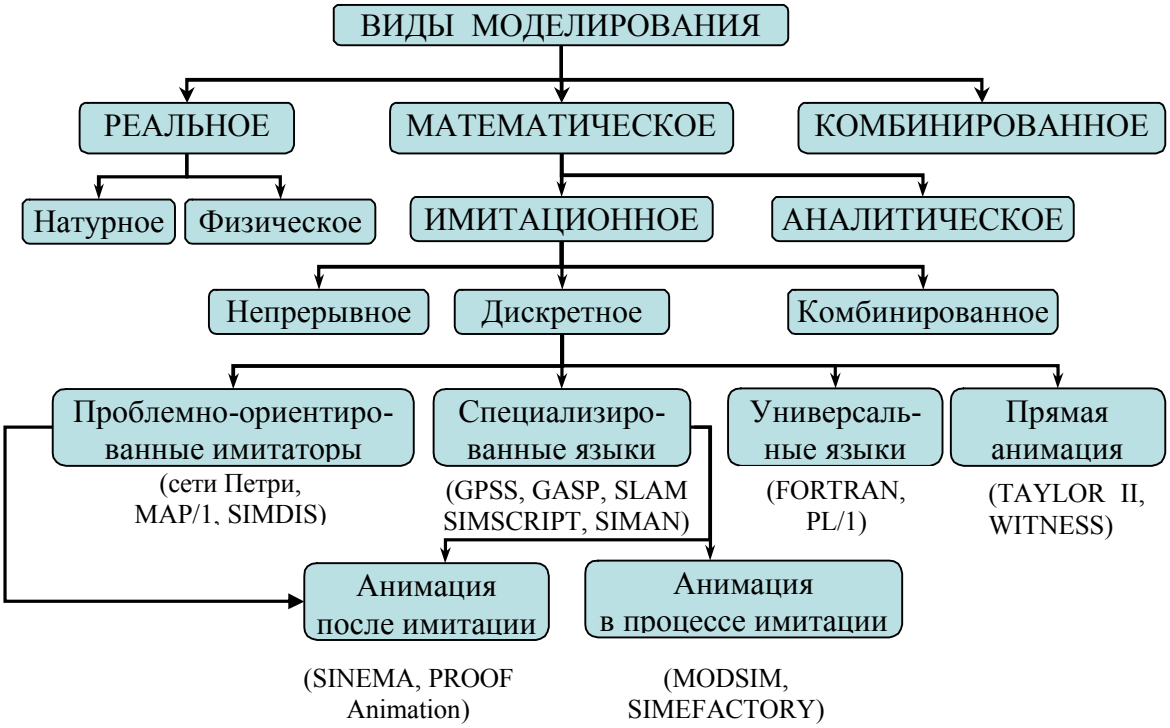


Рис. 1. Классификация видов моделирования технологических систем

С позиции математического моделирования технологический процесс можно описать либо при помощи системы уравнений, либо с помощью кружочков, точек, стрелок, либо закодировать при помощи команд какого-либо языка программирования, например Паскаль, C, Delphi. Несмотря на это разнообразие способов описания, результатом любого из них будет математическая модель. Следовательно, математическая модель – совокупность математических объектов (чисел, переменных, матриц, множеств, точек, отрезков прямых и т.д.) и отношений между ними, отражающая некоторые свойства моделируемого физического объекта, интересующие инженера-исследователя. Методы математического моделирования делят на аналитические и имитационные. Аналитические методы описывают объект в целом и сводятся к поиску оптимальных характеристик объекта. Аналитическая модель – математическая модель, представляющая собой совокупность аналитических выражений и зависимостей, позволяющих оценить определенные свойства моделируемого объекта. Во многих случаях невозможно получить аналитические зависимости, отображающие поведение и взаимосвязь элементов системы. Особенно трудно учесть действие случайных

факторов и динамику функционирования объекта, поэтому используют имитационное моделирование технологических процессов. Имитационная модель – математическая модель, отражающая поведение моделируемого объекта при заданных меняющихся во времени внешних воздействиях. Сущность имитационного моделирования состоит в искусственном воспроизведении в компьютере технологической системы с помощью специально построенной математической модели, хранящейся в памяти ЭВМ (рис. 2).

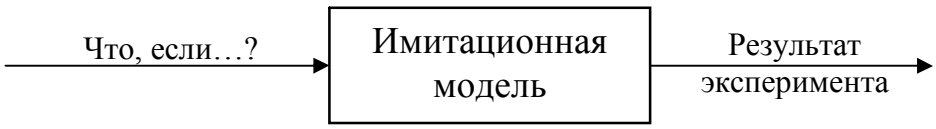


Рис. 2. Принцип имитационного моделирования

Над моделью проводят множество экспериментов типа «Что, если…?». Изменяя исходные показатели в модели, выбирают и реализуют на практике лучший вариант организации технологического процесса.

Имитационное моделирование применяют, если [2]:

- сложная динамика процесса не может быть описана системой уравнений;
- необходимо оценить влияние большого числа случайных факторов на показатели процесса;
- реальный процесс еще не создан;
- надо проверить варианты разрабатываемого процесса;
- неясно поведение системы за очень короткий или очень длинный период времени.

Имитационное моделирование может быть непрерывным, дискретным и дискретно-непрерывным (смешанным или комбинированным). Дискретное моделирование характеризуется конечным числом значений переменных системы. Функционирование дискретной модели можно представить как последовательную смену ее состояний в дискретные моменты времени, между которыми характеристики модели не изменяются. Например, при моделировании обработки детали в модели возникают события начала обработки и ее окончания. Все, что происходит между этими событиями, не учитывается. Непрерывное моделирование характеризуется бесконечным множеством значений переменных. Например, при моделиро-

вании движения жидкости по трубопроводу с течением времени через его сечение проходит какое-то количество жидкости, следовательно, в модели за любой самый малый отрезок времени значения переменных изменяются. Чтобы отобразить влияние переменных друг на друга, используют дифференциальные уравнения. Дискретно-непрерывное моделирование характеризуется наличием как дискретных, так и непрерывных значений переменных системы. Дискретно-непрерывные системы характеризуются тем, что в промежутках между дискретными состояниями система ведет себя как непрерывная. Например, при моделировании работы бензоколонки дискретное моделирование можно применить для отображения прихода бензовозов, которые пополняют емкости с горючим. А для того, чтобы исследовать динамику заполнения емкости с бензином, можно использовать непрерывное моделирование. Большинство технологических процессов сводится к дискретным с конечным числом состояний: начало и окончание обработки, погрузка и разгрузка груза, движение транспорта между пунктами погрузки и разгрузки. Поэтому остановимся на дискретном моделировании.

При построении имитационной модели для ввода ее в ЭВМ необходимо использовать языки программирования. Часто имитационные модели сложных технологических систем разрабатывают с использованием универсальных языков программирования (Паскаль, C, Delphi и т.п.). Процесс описывают последовательностью уравнений, которые кодируют в терминах используемого языка и вводят программу в компьютер (рис. 3).

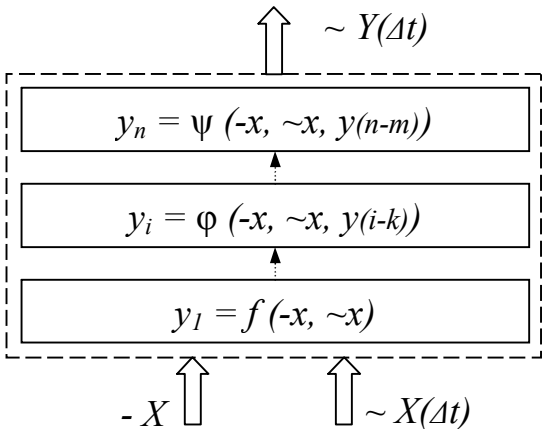


Рис. 3. Имитация техпроцессов с помощью универсальных языков программирования

В имитационных экспериментах изменяют исходные данные при случайных коэффициентах в уравнениях и получают технологические показатели производства. Такой подход требует аналитического описания процессов с последующим переводом полученной системы уравнений в программу для ЭВМ. Поэтому разработка таких программ занимает несколько человеко-месяцев труда специалистов по технологии, программированию и математике. Модель содержит сотни блоков, трудно поддается доработке (как правило, исправить или дополнить моделирующую программу может только тот, кто ее разрабатывал). Часто время разработки модели отстает от развития производства и модель становится ненужной. Альтернативой является использование методов, основанных на специализированных языках и системах имитационного моделирования, таких как GPSS, SIMAN, SLAM, SIMSCRIPT, SIMULA, GASP и др. (рис. 4).

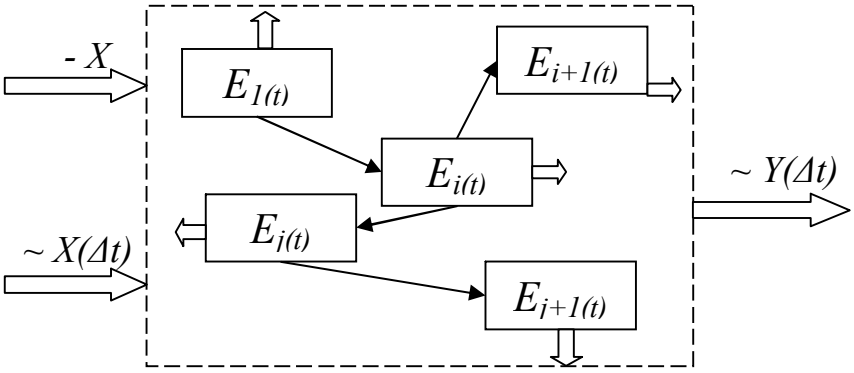


Рис. 4. Имитация техпроцессов
с помощью специализированных языков

Поведение системы в таких языках описывается от события к событию, означающим начало или окончание технологической операции. Процесс отображается не системой уравнений, а взаимодействием элементов производства E_1, ..., E_n во времени и пространстве.

Выделим основные преимущества использования специализированных языков компьютерной имитации:

- упрощение и ускорение создания имитационных моделей, так как отдельным операторам специализированного языка соответст-

вуют достаточно крупные блоки программного обеспечения, насчитывающие десятки-сотни операторов универсального языка;

- автоматическое формирование определенных типов данных, необходимых в процессе имитационного моделирования. В специализированных языках без специального на то указания пользователя собирается множество статистических данных, описывающих поведение модели. Пользователю нет необходимости создавать дополнительные подпрограммы для сбора и накопления этих данных;

- возможность конструирования сложных имитационных моделей пользователями, не являющимися профессиональными программистами. Программы имитационных моделей на специализированных языках моделирования близки к описаниям моделируемых систем на естественном языке. Например, последовательность технологических операций включить станок – обработать заготовку – выключить станок на специализированном языке будет выглядеть примерно так: занять прибор (команда SEIZE), задержать на время обслуживания (команда ADVANCE), освободить прибор (команда RELEASE);

- возможность моделирования систем без получения аналитических закономерностей процессов, так как техпроцесс отображается не системой уравнений, а взаимодействием отдельных динамических элементов во времени и пространстве.

Как альтернатива специализированным языкам развиваются проблемно-ориентированные имитаторы (сети Петри, MAP/1, SIMDIS, MAST). В проблемно-ориентированных имитаторах предусмотрены стандартные формы для ввода структуры и параметров объекта моделирования. После заполнения форм имитатор отображает процесс по шагам модельного времени. После этого анализируют поведение объекта во времени.

Недостатком имитационного моделирования является сложность технологической интерпретации результатов имитационных экспериментов. Например, в языке GPSS/H формируется стандартный файл отчета моделирования, информация в котором представлена в специфическом формате, и для оценки результатов ее необходимо интерпретировать. В имитаторе сетей Петри выводится матрица текущей разметки, которую необходимо расшифровывать. Так же если модель достаточно сложна, то разработчику трудно выявлять в ней ошибки. Поэтому к языкам имитации добавляют ани-

мацию, которая позволяет отображать динамику техпроцесса на мониторе. Соединяя языки компьютерной имитации с анимационными программами, создают программные комплексы – системы имитационного моделирования. Например, к GPSS и SLAM добавляют Proof Animation; CIMAN, соединенный с CINEMA, образует комплекс, называемый ARENA. При помощи анимации технолог, не владеющий программированием, может вводить характеристики оборудования, менять масштаб участков изображения, выводить на экран статистические показатели, быстро перемещаться во времени с целью анализа и прогноза ситуаций.

2. СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ

В основе множества специализированных языков компьютерной имитации (GPSS/H, SLAM, ARENA) лежит математический аппарат систем массового обслуживания (СМО). Теория массового обслуживания предназначена для формализации процессов функционирования систем, которые по своей сути являются процессами обслуживания. В качестве процесса обслуживания могут быть представлены различные по своей физической природе процессы функционирования производственных, технических, информационных, экономических и многих других систем. Например, заявки на обработку различных заготовок, потоки деталей и комплектующих изделий на сборочном конвейере цеха, потоки поставок продукции некоторому предприятию и др. При этом характерным для работы таких объектов является случайное появление заявок на обслуживание и завершение обслуживания в случайные моменты времени, т.е. стохастический характер процесса их функционирования.

Система массового обслуживания описывается потоком заявок, механизмом обслуживания, вместимостью системы и дисциплиной обслуживания. Эти атрибуты СМО более подробно описаны в пособии [3].

Пример простой СМО изображен на рис. 5.

Источник заявок (требований) – формирует входной поток, задерживая на какой-то отрезок времени поступление заявки в его состав. Входной поток – временная последовательность событий на входе СМО, для которой появление события подчиняется вероятно-

стным или детерминированным законам. Из входного потока заявки поступают на вход блока очереди. Блок очереди (или просто очередь) в соответствии с заданным вероятностным (или детерминированным) законом осуществляет выборку (или перераспределение) во времени событий во входном потоке для выдачи их на вход прибора обслуживания. Алгоритм постановки требований в очередь называют правилом формирования очереди, а алгоритм взаимодействия обслуживающих приборов с очередью – дисциплиной обслуживания. Прибор обслуживания осуществляет задержку во времени каждого поступившего на его вход события в соответствии с заданным детерминированным или случайным законом обслуживания. Обслуженная заявка с некоторой задержкой относительно поступившей на вход прибора поступает в выходной поток. Выходной поток отличается от входного в зависимости от правила формирования очереди и дисциплины обслуживания.

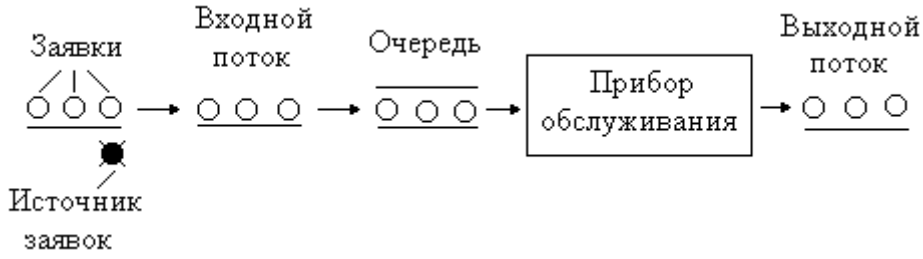


Рис. 5. Система массового обслуживания

Для СМО характерно, что все явления описываются с помощью событий, которые появляются в те или иные моменты времени, т.е. событиями могут являться: начало и окончание обработки, перемещение транспортного средства от одной точки до другой и т.д. Идеализация СМО заключается в том, что все остальные свойства реальных систем, которые не вписываются в эту модель событий, не учитываются (например, коэффициенты трения, изменение скорости при движении транспортного средства и т.д.).

В зависимости от соотношения времени между поступлением заявок и временем обслуживания заявок возможна одна из трех ситуаций:

- скорость обслуживания меньше скорости поступления заявок и перед прибором образуется очередь;

- скорость обслуживания равна скорости поступления заявок;

- скорость обслуживания больше скорости поступления заявок и прибор используется неполностью.

Таким образом, при отображении процесса в виде СМО, как правило, решаются три основные задачи:

- определение размера очереди заявок перед прибором;
- оценка времени обслуживания заявки несколькими приборами;
- оценка степени использования приборов.

2.1. Классификация систем массового обслуживания

Существует достаточно большое число различных моделей СМО. Рассмотрим их краткую классификацию, выделяя основные признаки [3].

1. По характеру источника заявок различают источники с конечным числом заявок и источники с бесконечным числом заявок. СМО с конечным числом заявок называются замкнутыми, а системы с бесконечным числом заявок – разомкнутыми. В первом случае в системе циркулирует конечное, обычно постоянное, число заявок. Заявки после завершения обслуживания возвращаются в источник, где они пребывают в течение некоторого времени, затем вновь поступают в систему. Во втором случае источник генерирует бесконечное число заявок, и работа источника никак не зависит от работы обслуживающей системы. Примером замкнутой СМО может служить процесс восстановления нескольких единиц оборудования несколькими ремонтниками (рис. 6).

Рис. 6. Структура замкнутой СМО

2. По отсутствию или наличию возможности ожидания для заявки выделяют системы с отказами и системы с ожиданием. В системах с отказами (с потерями) заявка, поступившая в момент, когда все обслуживающие приборы заняты, получает отказ, покидает систему и в дальнейшем процессе обслуживания не участвует. В системах с ожиданием (или без потерь) заявка, поступившая в момент, когда все приборы обслуживания заняты, становится в очередь и ожидает, пока не освободится один из приборов обслуживающей системы. Как только освобождается прибор, принимается к обслуживанию одна из заявок, стоящих в очереди. Системы с ожиданием делятся на системы с неограниченным ожиданием и системы с ограниченным ожиданием. В случае неограниченного ожидания любая заявка, рано или поздно пришедшая в систему, будет обслужена. В системах с ограниченным ожиданием накладываются те или иные ограничения: на длину очереди; на время пребывания заявки в очереди; на общее время пребывания заявки в системе.

3. По числу приборов СМО могут быть одноканальными (один прибор) или многоканальными (несколько приборов), а по числу этапов (фаз) обслуживания – однофазными или многофазными.

4. Дисциплина обслуживания в СМО может сводиться к четырем следующим видам:

- «первым пришел – первым обслужен»;
- циклическое обслуживание по одной заявке из различных источников;
- «последним пришел – первым обслужен»;
- приоритетная (заявка выбирается на обслуживание в соответствии с номером приоритета). Приоритет может быть абсолютным (если во время обслуживания приходит заявка с более высоким приоритетом, то обслуживание прерывается) и относительным (обслуживание происходит до конца, после чего обслуживается заявка с более высоким приоритетом).

При исследовании различных технологических процессов часто встречаются с прохождением заявок последовательно через несколько систем обслуживания. Примером может служить технологический процесс обработки деталей. Он содержит две стадии, на каждой из которых производится обработка деталей на соответствующей группе оборудования (рис. 7).

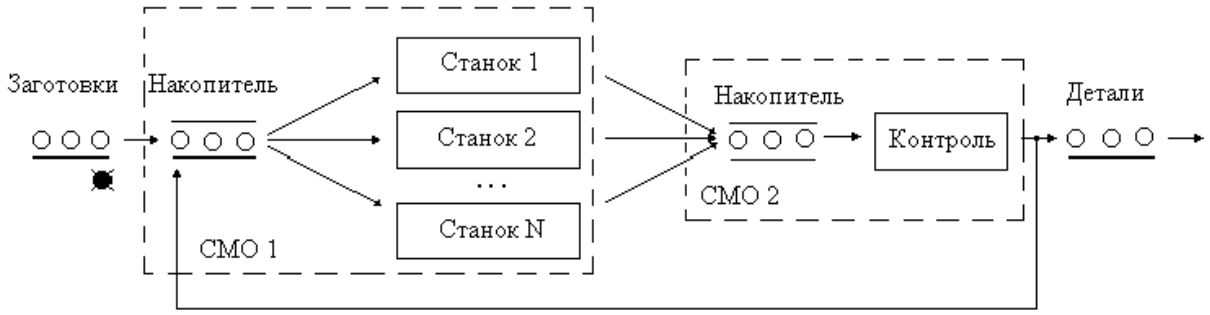


Рис. 7. Структура сети СМО

После второй СМО производится технический контроль, и детали поступают на дальнейшую обработку либо с некоторой вероятностью возвращаются на повторное выполнение операции. В этом случае СМО образуют сеть, которая характеризуется связями между отдельными СМО и свойствами самих систем.

Рассмотренная классификация СМО является далеко не полной и охватывает только основные их виды, используемые в качестве наиболее распространенных моделей систем.

Рассмотрим пример отображения функционирования гибкой производственной системы в терминах СМО и классифицируем ее.

Пример моделирования

Пусть в гибкой производственной системе (ГПС) детали последовательно обрабатываются двумя роботизированными технологическими комплексами (РТК). Первый РТК состоит из трех одинаковых станков и накопителя. Второй РТК состоит из одного станка и пристаночного накопителя. Заготовки через случайные интервалы времени поступают в ГПС (в накопитель РТК1). Время обработки заготовки каждым станком РТК1 также случайное. После обработки на РТК1 полуфабрикаты поступают на станок РТК2, обрабатываются в течение определенного времени и уходят на склад готовой продукции.

Метод построения модели

Будем считать, что поступающие в ГПС заготовки – это заявки в СМО. Тогда в соответствии с технологией эти заявки должны последовательно проходить через обслуживающие приборы РТК1 и РТК2, причем РТК1 содержит группу приборов, т.е. является многоканальным устройством с емкостью равной трем. Если все приборы обслуживания (РТК) заняты, то перед ними образуется очередь (рис. 8).

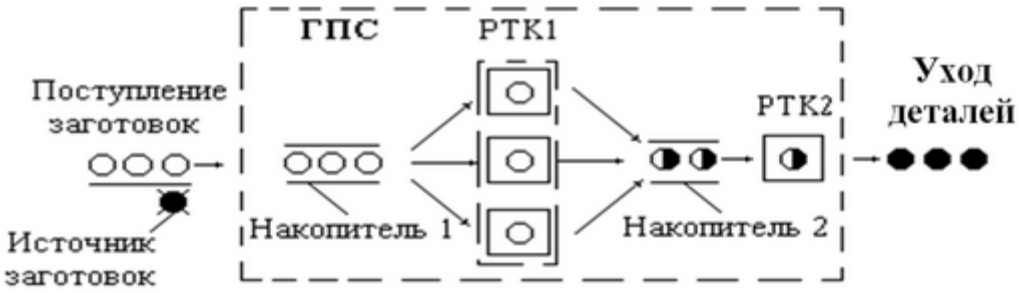


Рис. 8. ГПС в виде СМО

Исходя из этого работа ГПС может быть описана как многоканальная (несколько станков РТК1 могут обслуживать заявки одновременно), многофазная (заявки после обслуживания одним прибором переходят на обслуживание в следующий) СМО без потерь (заявкам разрешается ждать в очереди).

3. ТЕХНИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

В качестве программного обеспечения используется лицензионный комплекс языков компьютерной имитации GPSS/H и анимации Proof Animation, созданный фирмой Wolverine Software Corporation (США) в 1996 г. Комплекс имеет студенческую (Student GPSS/H – Student Proof Animation) и профессиональную (GPSS/H Professional – Proof Professional) версии.

Студенческая версия GPSS/H имеет некоторые ограничения:

- GPSS/H-модель не может содержать более 125 блоков;
- общее количество строк в программе не может превышать 250;
- объем динамической области памяти (COMMON) не может превышать 32 720 байта.

Если GPSS/H-программа написана так, что превышает любое из этих ограничений, то на экране после запуска программы появится сообщение об ошибках, указывающее, какие из трех ограничений были превышены. Например, если программа содержит более чем 125 блоков, на экране появится следующее сообщение:

ERROR: STUDENT-VERSION IS LIMITED TO 125 BLOCKS
LIMITS OF STUDENT-VERSION EXCEEDED - RUN TERMINATED

(ОШИБКА: СТУДЕНЧЕСКАЯ-ВЕРСИЯ ОГРАНИЧЕНА 125 БЛОКАМИ)

(ОГРАНИЧЕНИЕ СТУДЕНЧЕСКОЙ-ВЕРСИИ ПРЕВЫШАЕТ - ЗАВЕРШЕНИЕ)

Если вся область памяти (COMMON) будет занята, то на экране появится сообщение об ошибке 411:

ERROR NUMBER 411 - Out of COMMON: Add/change REALLOCATE Stmt? See release notes
(ОШИБКА 411 - «Превышение COMMON» - Добавить/изменить REALLOCATE?)

Студенческая версия Proof Animation также имеет некоторые ограничения:

- обработка специального файла, предназначенного для управления анимацией (.atf-файл), заканчивается после прочтения 1250 строк;
- файл, содержащий все нарисованные объекты, не может превышать 1250 байт;
- выполнение анимации прекращается через 120 секунд.

Для моделирования с использованием комплекса языков GPSS/H и Proof Animation требуется IBM – совместимый персональный компьютер класса 486 и выше, имеющий не менее 2 МБ свободной памяти на жестком диске и 4 МБ оперативной памяти, видеопамяти не менее 512 кБ. Может использоваться компьютер класса 386, снабженный математическим сопроцессором. Для работы с Proof Animation требуется Microsoft совместимая мышь. Комплекс может работать в операционной среде DOS, Windows 3.1 и выше, Windows NT, Windows 95 и выше.

4. МОДЕЛИРОВАНИЕ НА СПЕЦИАЛИЗИРОВАННОМ ЯЗЫКЕ GPSS/H

4.1. Общие сведения

Специализированный язык имитационного моделирования GPSS/H (General Purpose Simulation System) предназначен для отображения дискретных сложных систем различной физической природы. Язык основан на теории массового обслуживания – концепции движения объектов через приборы с некоторым временем обслуживания. Закон движения объектов и время обслуживания могут изменяться по различным законам распределения случайных величин.

Имитационную модель на языке GPSS/H сначала строят в виде блок-схемы, обеспечивающей наглядность перед записью программы. Блоки имеют свои графические интерпретации, с помощью которых отображается пространственная конструкция модели. На рис. 9 представлена гипотетическая блок-схема некоторой GPSS/H-модели.

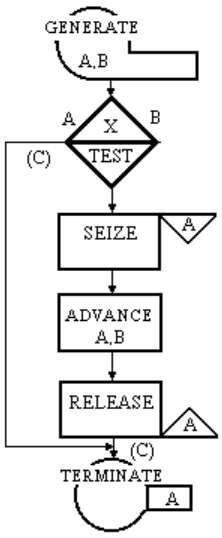


Рис. 9. Блок-схема GPSS/H-модели

Информация, представленная в блок-схеме, может относиться к трем различным категориям:

1. Местоположение. Каждый блок занимает определенное место в блок-схеме. Это место может быть определено нумерацией, которую интерпретатор осуществляет автоматически. Часто нужно

знать, какое место занимает тот или иной блок в модели. Это может оказаться необходимым для реализации ссылки из одного или более блоков на указанный блок. Для этих целей в GPSS/H используются символические имена, которые должны состоять не более чем из семи алфавитно-цифровых символов, причем первый символ должен быть алфавитным (русские буквы не допускаются).

2. Операции (ADVANCE, RELEASE и т.д.) – команды, описывающие основное функциональное назначение блоков.

3. Операнды (A,B,C,D,E...) – информация, специфичная для действия блока.

Исходная программа на языке GPSS/H, как и программа на любом языке программирования, представляет собой последовательность командных строк, которые вводятся в ЭВМ в следующем формате (рис. 10):

Рис. 10. Формат GPSS/H-программы

Символическое имя предназначено для ссылки на данную строку из другого места модели. Если такие ссылки отсутствуют, то имя указывать не обязательно. В поле операции записывается название команды, указывающее конкретную выполняемую функцию. В поле операндов заносится информация, уточняющая и конкретизирующая выполнение функции, определенной в поле операции. Необязательные комментарии предназначены для пояснения действия данной команды.

Командные строки записываются, начиная с первой позиции, в свободном формате, т.е. отдельные поля разделяются произвольным количеством пробелов. Программу пишут в любом текстовом редакторе, создающем неформатируемый текст в формате ASCII (NC,

VC, Far, Блокнот, WordPad и т.п.). Она должна иметь имя из латинских букв и расширение .gps (например, primer1.gps). Все буквы, кроме комментариев, должны быть заглавными.

GPSS/H-модель состоит из различных объектов. Для упрощения разделим эти объекты на три типа: транзакты, блоки и операторы.

Транзакты – динамические элементы GPSS/H-модели, представляют собой аналоги заявок в СМО. Они могут описывать, например, заготовки, транспортные средства, рабочих, требования на выполнение какой-либо операции и т.п. Блоки задают логику функционирования модели системы и определяют пути движения транзактов по модели. Блоки – аналоги приборов в СМО, которые отображают, например, участки цеха, станки, транспортные средства. Перемещаясь от блока к блоку, транзакты имитируют процессы, происходящие в системе: обработка заготовки, перемещение транспортного средства, восстановление вышедшего из строя станка и т.д. Блоки функционируют только тогда, когда в них находятся транзакты.

Операторы предназначены для управления процессом моделирования (прогоном модели), задания функций и последовательностей псевдослучайных чисел генераторов GPSS/H. Операторы напрямую не взаимодействуют с транзактами.

В процессе моделирования интерпретатор GPSS/H автоматически регистрирует и корректирует некоторую информацию, касающуюся различных элементов, используемых в моделях. Кроме информации, которая выдается по окончании моделирования, существует и такая информация, которая доступна в процессе моделирования. Эту информацию можно использовать с помощью атрибутов модели. Атрибутами являются: состояние прибора (занят, не занят), счетчик циклов занятий прибора, коэффициент использования прибора, среднее время задержки на одно занятие, величина параметра транзакта, время пребывания транзакта в модели, генераторы случайных чисел. В процессе моделирования системы транзакты взаимодействуют с блоками, в результате чего происходят изменения их атрибутов, а также преобразования арифметических или логических значений. Такие преобразования называются событиями.

4.2. Моделирование начала техпроцесса

Принцип работы GPSS/H-модели заключается в перемещении транзактов от блока к блоку. Вначале в модели нет ни одного транзакта. Следовательно, для того, чтобы модель техпроцесса начала функционировать, необходимо создать и запустить транзакты в модель. В GPSS/H для этих целей используется блок GENERATE.

Блок GENERATE (генерировать) – создание и ввод транзактов в модель (рис. 11).

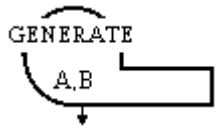


Рис. 11. Блок GENERATE

Блок имеет 8 операндов (A,B,C,D,E,F,G,H). Для начала рассмотрим первые пять.

Примечание

Транзакты могут создаваться через интервалы времени, распределенные по различным законам. Все возможные виды распределения временных интервалов разделим на равномерное и все другие виды распределения. И до п. 4.8. Моделирование случайных событий будем рассматривать только равномерное распределение.

В операнде A записывают среднее значение равномерно распределенных интервалов времени, в операнде B – половину поля допуска распределенного интервала. Операнд C предназначен для задания смещения интервалов. В операнде D указывают значение, ограничивающее число создаваемых транзактов. В операнде E задают приоритет вышедшим транзактам. По умолчанию значения операндов A, B, E равны нулю, а операнда D – ∞. Операнд C по умолчанию не задает смещение.

Примеры блока GENERATE:

GENERATE 4.1,2.1

транзакты создаются каждые 4.1±2.1 единицы времени.

GENERATE 4.1,,3

первый транзакт создается во время, равное 3, второй во время – 7.1, третий – 11.2 и т.д.

GENERATE 1,,,1

первый и единственный транзакт создается во время, равное 1.

GENERATE 8,1,,,,4

каждый созданный транзакт будет иметь уровень приоритета, равный 4 (транзакт с более высоким приоритетом обслуживается раньше, чем транзакт с меньшим приоритетом).

Операнды F, G, H будут рассмотрены позже.

При построении модели необходимо учитывать, что нельзя последовательно ставить два блока GENERATE и направлять транзакты в этот блок. Например, последовательность блоков

GENERATE ,,, 1

GENERATE 4.1

вызовет ошибку в логике работы модели.

В модели может быть несколько блоков GENERATE, каждый такой блок образует так называемый сегмент модели. В студенческой версии GPSS/H можно использовать 125 сегментов.

4.3. Моделирование завершения техпроцесса

Созданные транзакты, двигаясь по модели, имитируют операции, происходящие в реальном техпроцессе. Для того чтобы остановить моделирование, необходимо вывести определенное количество транзактов из модели или остановить модель через какое-то заданное время. Для этого используют блок TERMINATE.

Блок TERMINATE (завершить) – удаление транзактов из модели (рис. 12).

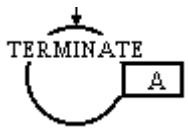


Рис. 12. Блок TERMINATE

При входе транзакта в блок TERMINATE он уничтожается и из специального счетчика завершения вычитается целое число, записанное в операнде A. Счетчик завершения – ячейка памяти, которая хранит положительное целое значение, записанное в начале моделирования. Счетчик завершения в GPSS/H-модели может быть

только один. Если его значение станет равным 0 (или меньше 0), моделирование прекратится. Если значение операнда A не установлено (т.е. равно 0), значение счетчика завершения не изменяется.

Примеры блока TERMINATE:

TERMINATE 1

удаляет транзакты из модели, при этом содержимое счетчика завершения (каждый раз при входе транзакта) уменьшается на единицу.

TERMINATE

удаляет транзакты из модели, при этом содержимое счетчика завершения не изменяется.

После того как написана GPSS/H-программа, прежде чем выполнять прогоны модели, к ней необходимо добавить специальные операторы. Одним из таких операторов является START.

Оператор START – устанавливает значение счетчика завершения.

В операнде A оператора START записывается начальное значение счетчика (больше 0).

Примеры оператора START:

1. Предположим, что требуется промоделировать техпроцесс в течение 480 единиц времени. Для этого необходимо:

а) в модель включить сегмент из двух блоков, называемый счетчиком модельного времени (рис. 13);

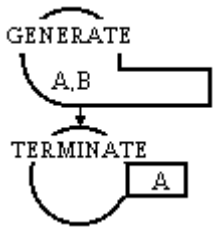


Рис. 13. Счетчик модельного времени

б) во всех прочих блоках TERMINATE операнд A обнулить;

в) операнд A оператора START установить равным 1.

Часть GPSS/H-программы будет иметь следующий вид:

```
* 1-й сегмент модели
GENERATE     3.2,1.2,,2
. . .
```

```
TERMINATE

* 2-й сегмент модели
GENERATE     480
TERMINATE    1
START     1
```

В момент модельного времени 480 транзакт выйдет из блока GENERATE второго сегмента и сразу же попадет в блок TERMINATE. Поскольку операнд этого блока содержит 1, то из счетчика завершения отнимется 1. Это уменьшит значение счетчика до 0, и интерпретатор остановит прогон модели.

2. Предположим, что требуется закончить прогон модели после того как будет обслужено 100 транзактов. Для этого необходимо:

а) установить значение операнда A оператора START равным 100;

б) установить операнд A блока TERMINATE равным 1.

Часть GPSS/H-программы будет иметь следующий вид:

```
GENERATE     3.2,1.2,,2
. . .
TERMINATE    1
START     100
```

Когда транзакт будет входить в блок TERMINATE, счетчик завершения будет уменьшаться на 1. Прогон будет продолжаться, пока значение счетчика не достигнет 0 (т.е. пока не удалится 100 транзактов).

4.4. Моделирование технологических операций

Операции в технологических процессах (обработка заготовок, транспортировка изделий и т.п.) отображаются в СМО временем обслуживания заявки прибором. Для имитации этого в GPSS/H-модели необходимо задержать транзакт в приборе на время обслуживания заявки. Для этих целей используют блок ADVANCE.

Блок ADVANCE (задержать) – задержка транзактов (рис. 14).

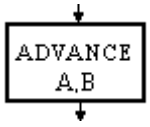


Рис. 14. Блок ADVANCE

При равномерном распределении в операнд A записывают среднее время задержки транзакта, а в операнд B – половину поля допуска. (Неравномерное распределение будет рассмотрено позже). Значения по умолчанию для операндов A и B равны 0.

Примеры блока ADVANCE:

ADVANCE 2.1

транзакты задерживаются на 2.1 единицы времени.

ADVANCE 3.75,1

задержка транзактов колеблется в интервале от 2.75 до 4.75 единиц.

В блоке ADVANCE может одновременно находиться более одного транзакта. При этом каждый транзакт задерживается на определенное в блоке время.

4.5. Моделирование технологического оборудования

При формализации технологических процессов при помощи математического аппарата СМО элементы, которые представляют обслуживание, моделируются приборами. Прибором могут быть, например, станок, транспортное средство, робототехнический комплекс и т.п. Для их моделирования в GPSS/H используется пара блоков SEIZE и RELEASE.

Блок SEIZE (занять) – занятие прибора обслуживания (рис. 15).

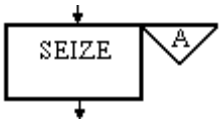


Рис. 15. Блок SEIZE

В операнде A записывают имя занимаемого прибора (указывается обязательно). Если транзакт входит в блок SEIZE, он занимает прибор обслуживания и закрывает вход следующим транзактам.

Пример блока SEIZE:

SEIZE SERVER

при входе транзакта в блок занимается прибор с именем SERVER.

Блок RELEASE (освободить) – освобождение занятого прибора обслуживания (рис. 16).

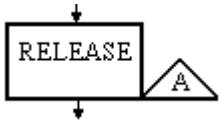


Рис. 16. Блок RELEASE

Операнд A – имя освобождаемого прибора (указывается обязательно).

Пример блока RELEASE:

RELEASE SERVER

при входе транзакта прибор SERVER освобождается и разрешается вход в него (в блок SEIZE SERVER) других транзактов.

Приборы в GPSS/H-моделях могут быть размещены по разным схемам.

Схема «Классическая»

SEIZE BOX
ADVANCE 16,4
RELEASE BOX

Транзакт занимает прибор BOX, задерживается в нем на 16±4 единицы времени и освобождает его.

Схема «Прибор в приборе».

SEIZE SERVER
ADVANCE 3.0,0.5
SEIZE BOX
ADVANCE 0.1
RELEASE BOX
RELEASE SERVER

После задержки транзакта в блоках ADVANCE приборы BOX и SERVER освобождаются одновременно.

Блок RELEASE не запрещает вход транзактам. Если в модели делается попытка освободить свободный прибор или если транзакт пытается освободить прибор, занятый другим транзактом, то интер-

претатор GPSS/H выдаст сообщение об ошибке и прогон модели прекратится. Такая ошибка не будет происходить в схеме «Прибор в приборе» начиная с первого транзакта, который входит внутрь комбинации блоков SEIZE-RELEASE, моделирующих занятие прибора SERVER. Однако при разработке более сложных моделей, когда транзакт пытается освободить прибор, который был занят другим транзактом или не был занят вовсе, это является распространенной ошибкой [4].

Схема «Использование прибора несколько раз»

```
SEIZE       BOX
ADVANCE     34.4,2.3
RELEASE     BOX
ADVANCE     2,3
SEIZE       BOX
ADVANCE     12,3
RELEASE     BOX
```

Транзакт, проходя по модели, занимает и освобождает прибор BOX два раза.

Пример моделирования

Рассмотренные блоки GENERATE, TERMINATE, ADVANCE, SEIZE, RELEASE и оператор START уже позволяют строить простейшие модели технологических процессов. Создадим модель работы робототехнического комплекса (РТК).

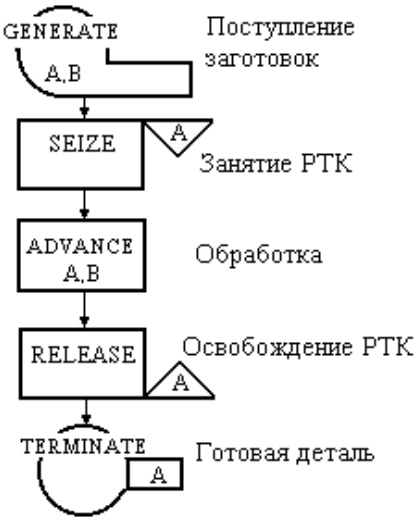
Заготовки поступают на вход РТК через случайные интервалы времени, после чего они либо сразу же обрабатываются, если РТК свободен, либо становятся в очередь. В РТК одновременно можно обрабатывать только одну заготовку. Время обработки каждой заготовки случайно. Примем допущение, что интервалы между поступлениями заготовок и времена их обработки имеют равномерное распределение со значениями 100±40 и 80±50 секунд соответственно. Необходимо построить модель работы РТК и определить его коэффициент загрузки.

Примем за единицу модельного времени одну секунду реального времени и допустим, что для получения статистически достоверных оценок коэффициента загрузки РТК достаточно 100 измерений. Тогда GPSS/H-модель РТК может быть построена в виде последовательности блоков и команд, представленных в табл. 1.

В данной модели прибор с именем RTK имитирует работу робототехнического комплекса. Транзакты создаются блоком GENERATE до тех пор, пока через блок TERMINATE не пройдет 100 транзактов.

Таблица 1

Модель роботизированного технологического комплекса (вариант 1)

Блок-схема GPSS/H-модели	GPSS/H-программа
	GENERATE 100,40 SEIZE RTK ADVANCE 80,50 RELEASE RTK TERMINATE 1 START 100

4.6. Моделирование группы технологического оборудования

Часто в технологических системах используют группу оборудования. В терминах СМО – группу приборов, каждый из которых выполняет аналогичное обслуживание, например, параллельно работающие токарные или фрезерные станки на участке механообработки некоторого предприятия. Такие приборы, обладающие одинаковыми свойствами, моделируют многоканальным устройством. Число приборов, которое моделирует многоканальное устройство, определяется пользователем и называется емкостью многоканального устройства.

Так же как для приборов обслуживания, для моделирования многоканального устройства используются два блока ENTER и LEAVE.

Блок ENTER (войти) – занятие прибора(ов) из группы (рис. 17).

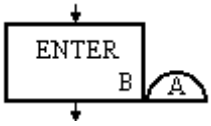


Рис. 17. Блок ENTER

Имитирует включение параллельно работающего оборудования. При входе транзакта в блок ENTER моделируется занятие приборов, число которых указывается в операнде B. Имя группы параллельно работающих приборов указывается в операнде A.

Примеры бока ENTER:

ENTER TOKARN

транзакт занимает один прибор многоканального устройства TOKARN.

ENTER TOOLS,2

транзакт занимает два прибора многоканального устройства TOOLS (оба модуля должны быть свободны).

Блок LEAVE (выйти) – освобождение прибора(ов) из группы (рис. 18).

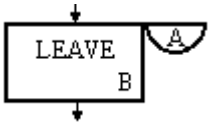


Рис. 18. Блок LEAVE

Имитирует выключение параллельно работающего оборудования. Значения операндов A и B те же, что и для блока ENTER.

Примеры блока LEAVE:

LEAVE TOKARN

транзакт освобождает один прибор многоканального устройства TOKARN.

LEAVE TOOLS, 2

транзакт освобождает два прибора многоканального устройства TOOLS.

Многоканальные устройства характеризуются емкостью, которую необходимо задавать в начале моделирования. Это делается при помощи оператора STORAGE.

Оператор STORAGE – задает емкость многоканального устройства.

В операнде A указывается имя многоканального устройства, а в операнде B – целое число, определяющее его емкость.

Примеры оператора STORAGE:

STORAGE S(TOKARN),2

задается многоканальное устройство с именем TOKARN и с пропускной способностью, равной 2.

STORAGE S(KOMP),3/S(FAX),4

задаются многоканальные устройства KOMP с емкостью 3 и FAX с емкостью 4.

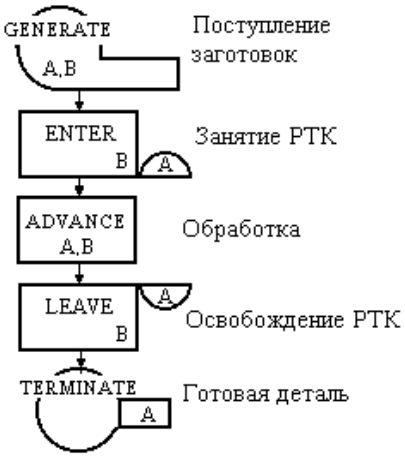
Пример моделирования

Блоки ENTER и LEAVE позволяют моделировать технологические процессы, отображаемые многоканальными СМО. Занятие канала в течение некоторого времени (обслуживание в СМО) отображается блоком ADVANCE, который помещается между блоками ENTER и LEAVE.

В качестве примера использования блоков ENTER и LEAVE изменим условие рассмотренной выше модели роботизированного технологического комплекса. Предположим, что в РТК одновременно можно обрабатывать три заготовки. Для отображения этого условия используем многоканальное устройство с емкостью 3. По сравнению с моделью, рассмотренной выше, блоки SEIZE и RELEASE заменены на блоки ENTER и LEAVE (табл. 2).

Таблица 2

Модель роботизированного технологического комплекса (вариант 2)

Блок-схема GPSS/H-модели	GPSS/H-программа
	STORAGE S(RTK),3 GENERATE 100,40 ENTER RTK ADVANCE 80,50 LEAVE RTK TERMINATE 1 START 100

4.7. Сбор статистики о накопителях

При использовании в GPSS/H-моделях приборов и многоканальных устройств интерпретатор автоматически собирает информацию относительно коэффициентов загрузки приборов, среднего времени занятия приборов и т.п. Очень часто бывает необходимо собрать некоторую статистику о накопителях (об очереди): максимальное и среднее значения длины очереди, среднее время ожидания в очереди и т.д. Специализированный язык GPSS/H имеет специальные блоки QUEUE и DEPART, позволяющие осуществлять автоматический сбор такого рода статистической информации.

Блок QUEUE (встать в очередь) – начало автоматического сбора статистических данных о накопителях (рис. 19).

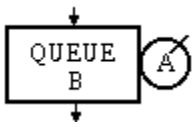


Рис. 19. Блок QUEUE

В операнде A записывается имя очереди, к которой необходимо присоединиться, а в операнде B – число элементов, на которое должно измениться значение содержимого очереди.

Блок DEPART (покинуть очередь) – окончание автоматического сбора статистических данных о накопителях (рис. 20).

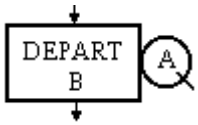


Рис. 20. Блок DEPART

Значения операндов A и B те же, что и для блока QUEUE.

Ниже приводятся GPSS-модели роботизированного технологического комплекса, в которых используются блоки QUEUE и DEPART. В варианте *а* измеряется время нахождения транзактов в очереди перед блоком SEIZE (очередь с именем COM), в варианте *б* – общее время пребывания транзакта в приборе обслуживания с

учетом времени нахождения транзакта в очереди (очередь с именем VS):

а

```
GENERATE    100,400
QUEUE   COM
SEIZE       RTK
DEPART  COM
ADVANCE     80,50
RELEASE     RTK
TERMINATE   1
START       50
```

б

```
GENERATE    100,40
QUEUE   VS
SEIZE       RTK
ADVANCE     80,50
RELEACE     RTK
DEPART  VS
TERMINATE   1
START       50
```

В этих моделях помимо информации о времени моделирования и о приборах будет собираться информация об очереди, которая включает: максимальный и средний размер очереди за время моделирования; общее число транзактов, которые занимали очередь; среднее время нахождения транзакта в очереди и т.п.

4.8. Моделирование случайных событий

Процессы в реальных технологических системах, как правило, случайны. Поэтому случайные числа играют важную роль в процессе моделирования. Они используются для вычисления времени между двумя входами транзактов через блок GENERATE, вычисления времени задержки транзактов в блоке ADVANCE, определения вероятностной передачи транзактов через блок TRANSFER, вероятностной проверки условия в блоке TEST (см. ниже). Все эти вычисления и определения производятся в соответствии с функциями. Функции могут быть дискретные и непрерывные, детерминированные и вероятностные. Для розыгрыша случайных чисел при использовании вероятностных функций используются встроенные датчики равномерного распределения в интервале (0, 1). GPSS/H имеет 8 таких датчиков с именами RN1...RN8. Эти датчики являются датчиками псевдослучайных величин, получаемых с помощью некоторого алгоритма.

Для задания функций в GPSS/H используется оператор FUNCTION.

Оператор FUNCTION определяет функцию.

В поле имени записывается имя функции. В поле операций записывается слово FUNCTION. В операнде A записывается номер используемого генератора случайных чисел (от 1 до 8). В операнде B указывается DN или CN, что соответствует определению дискретной или непрерывной функций. N – число различных значений, получаемых случайной переменной (суммарная частота). Далее должны следовать строки определения функции со значениями суммарной частоты и соответствующими им значениями случайной переменной.

Ранее мы рассматривали только самый тривиальный закон распределения – равномерный. Если существует необходимость моделировать случайные процессы, распределенные по другим законам распределения, то необходимо задавать либо функции, определяемые пользователем, либо встроенные в GPSS/H законы распределения.

4.8.1. Определение дискретной функции

Дискретные функции предназначены для имитации дискретных случайных процессов, заданных функцией распределения *F(x)*. Функция распределения задается таблицей, в которой указаны пары: значения аргумента, имеющего равномерное распределение в интервале (0, 1), и соответствующие значения функции. Например, в табл. 3 задана дискретная случайная величина, принимающая значения 2 с вероятностью 0.15, значения 5 – с вероятностью 0.20 и т.д.

Таблица 3

Дискретная функция распределения

Значения случайной величины	Относительная частота	Суммарная частота
2	0,15	0,15
5	0,20	0,35
8	0,25	0,60
9	0,22	0,82
12	0,12	1,00

В соответствии с информацией таблицы можно задать дискретную функцию, определив суммарную частоту случайной переменной и использовав оператор FUNCTION следующим образом:

```
KAT FUNCTION      RN4,D5
.15,2/.35,5/.6,8/.82,9/1,12
```

Функция имеет символическое имя KAT. В качестве источника случайных чисел выступает RN4. Дискретная переменная может иметь пять значений. Суммарные частоты и соответствующие им пять значений записаны как пять пар чисел на следующей строке. На рис. 20 приведена графическая интерпретация этой функции.

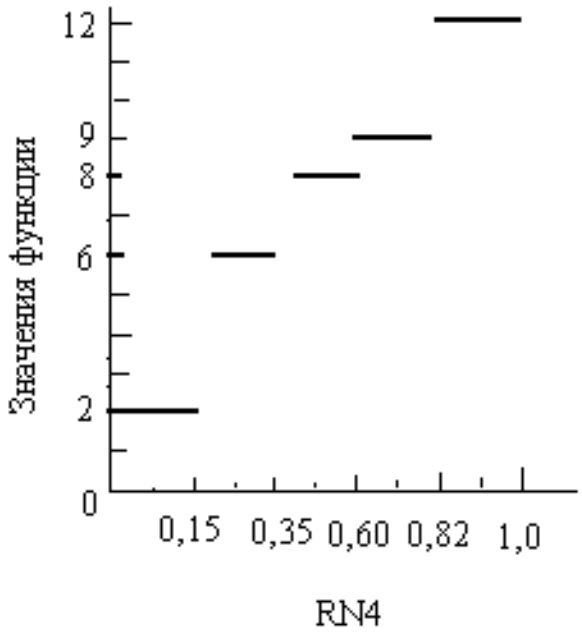


Рис. 20. Графическая интерпретация дискретной функции

Функция состоит из серии горизонтальных ступенек. Например, правая ступенька перекрывает значения до 0,15 включительно. Вторая ступенька начинается от значения 0,15 и продолжается до 0,35 включительно и т.д. На дискретную функцию можно ссылаться для розыгрыша значений в блоках GENERATE и ADVANCE.

Процедуру имитационного моделирования с использованием дискретной функции покажем на примере обработки заготовок различных типов станком с ЧПУ.

На станок с ЧПУ каждые 33±2 минуты (распределение равномерное) поступают заготовки различных типов – А и Б, которые требуют различного времени обработки. Заготовки типа А посту-

пают с вероятностью 0,65 и требуют для обработки 45 минут. Заготовки типа Б поступают с вероятностью 0,35, их время обработки составляет 30 минут.

Для реализации модели подобного техпроцесса воспользуемся дискретной функцией, принимающей значения 45 с вероятностью 0,65, и значения 30 с вероятностью 0,35. GPSS/H-модель техпроцесса приведена ниже.

```
RAND FUNCTION   RN1,D2    Определение дискретной функции
0.65,50/1,30              Строки определения функции
      GENERATE    33,2    Поступление заготовок
      SEIZE       STAN        Включение станка с ЧПУ
      ADVANCE     FN$RAND     Обработка
      RELEASE     STAN        Выключение станка с ЧПУ
      TERMINATE 1             Готовая деталь
      START     500           Программа выпуска
```

В приведенной модели функция, принимающая случайные значения с заданными вероятностями, имеет имя RAND. В модели показано, каким образом следует обращаться к этой функции для реализации изменяющейся задержки в блоке ADVANCE. Начало ссылки на функцию содержит символы «FN» (это общее название функций всех типов). Затем следует символ «$», после которого идет собственно символическое имя.

4.8.2. Определение непрерывной функции

Дискретные случайные переменные могут иметь только фиксированное число значений. В противоположность этому непрерывные случайные переменные могут иметь неограниченное число различных значений на заданном интервале. Для определения непрерывной функции, так же как и для определения дискретной, используют оператор FUNCTION. Непрерывные функции предназначены для имитации случайных процессов, заданных непрерывной функцией распределения *F(x)*. Функция распределения может быть также задана таблицей, в которой указаны пары: значения аргумента, имеющего равномерное распределение в интервале (0, 1), и соответствующие интервалы значений функции. Например, в табл. 4

заданы интервалы значений непрерывной случайной величины. Первая строка таблицы показывает, что ни одно значение случайной величины не попало в интервал от 0 до 15. Вторая строка таблицы показывает, что 7 % значений случайной величины попали в интервал от 15 до 30 и т.д.

Таблица 4

Непрерывная функция распределения

Интервалы значений случайной величины	Относительная частота попадания в интервал	Суммарная частота
Менее 15	0,00	0,00
от 15 до 30	0,07	0,07
от 30 до 45	0,25	0,32
от 45 до 60	0,41	0,73
от 60 до 75	0,19	0,92
от 75 до 90	0,08	1,00

Так же как и при определении дискретной функции, в соответствии с информацией табл. 4 можно задать непрерывную функцию, определив суммарную частоту случайной (см. табл. 4) и использовав оператор FUNCTION:

```
BOX FUNCTION      RN1,C6
.0,15/.07,30/.32,45/.73,60/.92,75/1,90
```

Функция, имеющая символическое имя BOX, будет принимать с вероятностью 0 – значения, равномерно распределенные в интервале (0, 15), с вероятностью, равной 0.07– значения, равномерно распределенные в интервале (15, 30), с вероятностью 0.32 – значения, равномерно распределенные в интервале (30, 45) и т.д. Последний интервал значение 90 не включает потому, что результат розыгрыша случайной величины, равномерно распределенной в интервале (0, 1) с помощью датчика RN1, никогда не будет равен 1 – все датчики с именами RN в GPSS/H дают значения, равномерно распределенные от 0.000001 до 0.999999. Графическая интерпретация функции представлена на рис. 22.

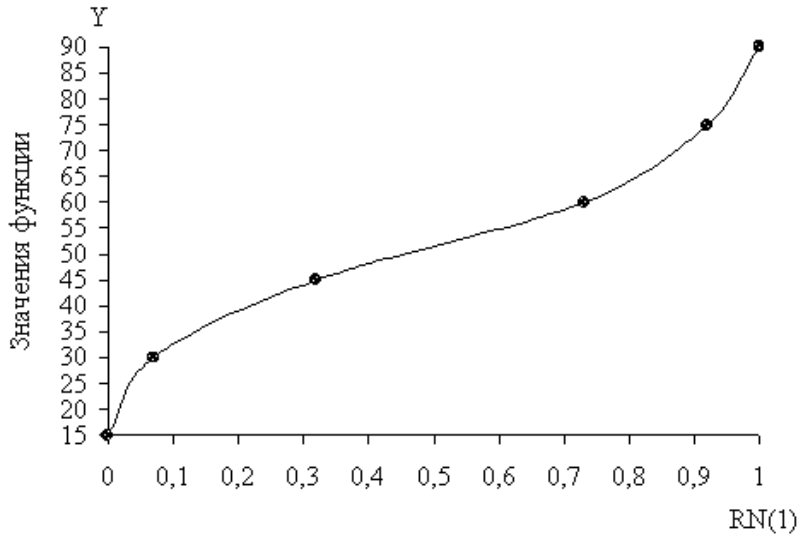


Рис. 22. Графическая интерпретация функции BOX

Если RN1 выдает число 0,07 или меньше, интерпретатор выполняет линейную интерполяцию между значениями 15 и 30 и определяет значение функции. Если RN1 в точности равно 0,07, значением функции будет 30 и т.д.

На непрерывную функцию можно ссылаться из блоков GENERATE и ADVANCE, так же как и на дискретную.

4.8.3. Определение функций по заданному закону распределения

Если закон распределения случайной переменной заранее известен, нет необходимости задавать функцию вручную, определяя значения случайной переменной и соответствующие значения суммарной частоты. Для моделирования случайных событий с известным законом распределения GPSS/H имеет встроенные функции, которые генерируют произвольные случайные переменные из 26 популярных распределений. В табл. 5 приведены определения двух наиболее применяемых законов распределения, поддерживаемых студенческой версией GPSS/H.

Таблица 5

Встроенные функции

Распределение	Формат функции
Нормальное	RVNORM(RN1, мат. ожидание, дисперсия)
Экспоненциальное	RVEXPO(RN1, параметр λ)

Например, необходимо смоделировать поступление заготовок в робототехнический комплекс. Интервалы между поступлением заготовок распределены согласно экспоненциальному закону с параметром $\lambda = 7$. Для имитации поступления заготовок используем блок GENERATE. Тогда в соответствии с этим условием и табл. 5, запишем:

```
GENERATE      RVEXPO(1,7)
```

4.9. Пример имитационного моделирования

Процедуру имитационного моделирования покажем на примере изготовления деталей разными типами станков.

Технологический процесс содержит две стадии. На каждой стадии производится обработка деталей на двух станках типа А и одном типа В. После обработки заготовки на станке типа А полуфабрикаты идут на дальнейшую обработку на станок типа В. Готовые детали поступают в накопитель.

Интервалы поступления заготовок на группу станков типа А распределены согласно экспоненциальному закону с параметром $\lambda = 28$ минут. Продолжительность обработки заготовки станком типа А составляет 63±9 минут (распределение равномерное), а станком типа В – 55±5 минут (распределение равномерное). Время транспортировки заготовки до группы станков типа А составляет 32±10 минут, а полуфабрикатов до станка типа В – 12±5 минут (распределение нормальное). Предполагается, что между двумя станциями существует неограниченная очередь.

Необходимо построить модель технологического процесса изготовления деталей, используя специализированный язык компьютерной имитации GPSS/H, и определить продолжительность изготовления 50 деталей, коэффициент загрузки группы станков типа А и станка типа В, среднее значение длин очередей перед группой станков типа А и станком типа В.

4.9.1. Метод построения модели

Построим блок-схему модели технологического процесса (рис. 23).

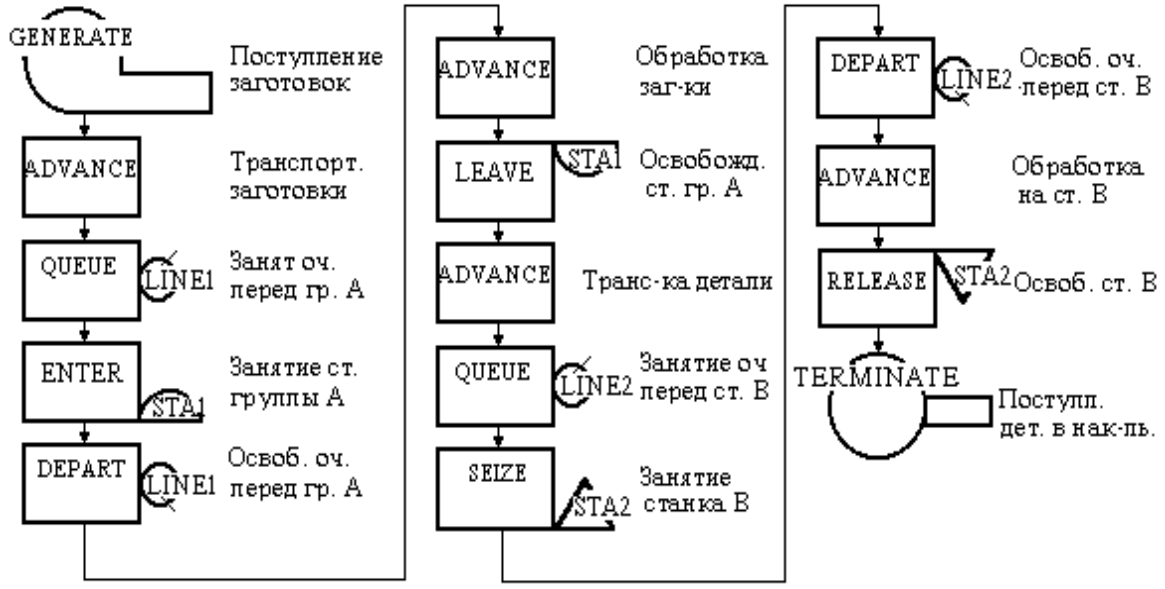


Рис. 23. Блок-схема имитационной модели технологического процесса изготовления деталей

Пусть транзакт представляет собой заготовку, которая по мере продвижения по модели преобразуется в готовую деталь. Поступление заготовок в соответствии с заданным распределением имитируем блоком GENERATE. Группу станков типа А моделируем многоканальным устройством STA1, т.е. двумя блоками ENTER и LEAVE, между которыми ставим блок ADVANCE для имитации обработки заготовки. Так как станок второй группы всего один, имитируем его прибором обслуживания STA2, т.е. двумя блоками SEIZE и RELEASE. Для сбора статистики об очередях перед станками типов А и В вводим два регистратора очереди посредством блоков QUEUE и DEPART. Транспортировку заготовки до станков типа А и станка типа В моделируем блоками ADVANCE. Блок TERMINATE вводим для имитации поступления готовой детали в накопитель.

4.9.2. Подготовка модели к запуску

Для ввода в ЭВМ в соответствии с блок-схемой пишем GPSS/H-программу, используя любой текстовый редактор, создающий неформатируемый текст (формат ASCII). Прежде чем запускать модель (осуществлять прогон), к GPSS/H-программе необходимо добавить два оператора SIMULATE и END.

Оператор SIMULATE – указывает интерпретатору, что должен быть осуществлен прогон модели (начальная команда в GPSS/H-программе). Если этот оператор отсутствует, то интерпретатор проверяет правильность записи модели на языке GPSS/H, но прогона модели не выполняет.

Оператор END – завершает работу интерпретатора.

Следовательно, для начала моделирования используем оператор SIMULATE. Для задания емкости многоканального устройства, имитирующего станки группы А, вводим оператор STORAGE. Оператор START вписываем для задания значения счетчика завершения, которое соответствует количеству требуемых деталей. Оператор END вводим для окончания моделирования. После перевода блок-схемы модели и добавления необходимых операторов GPSS/H-программа, подготовленная к запуску, будет выглядеть следующим образом:

```
SIMULATE                    Начало моделирования
STORAGE  S(STA1),2          Задание станков в группе А
GENERATE      RVEXPO(1,28)  Поступление заготовок
ADVANCE RVNORM(3,32,10)     Транспортировка заготовки
QUEUE         LINE1         Занятие очереди перед станками
                            группы А
ENTER         STA1          Занятие станка из группы А
DEPART   LINE1              Освобождение очереди перед
                            станками группы А
ADVANCE 63,9                Обработка на станке группы А
LEAVE         STA1          Освобождение станка группы А
ADVANCE RVNORM(5,12,5)      Транспортировка детали
QUEUE         LINE2         Занятие очереди перед станком
                            типа В
SEIZE         STA2          Занятие станка типа В
DEPART   LINE2              Освобождение очереди перед
                             станком типа В
ADVANCE 55,5                Обработка на станке типа В
RELEASE  STA2               Освобождение станка типа В
TERMINATE     1             Поступление деталей в накопи-
                            тель
START         50            Программа выпуска
END                         Окончание моделирования
```

После написания программы сохраним файл с расширением .gps (например, primer1.gps).

4.9.3. Запуск модели и получение результатов

Для осуществления прогона модели необходимо запустить интерпретатор путем загрузки файла gpssh.exe. После этого появится строка:

ENTER SOURCE FILE NAME: (Введите имя исходного файла)

На этот запрос необходимо ввести имя GPSS/H-программы (в нашем случае, primer1) и нажать ENTER. После этого интерпретатор проверит модель на наличие ошибок и запустит ее. После прогона GPSS/H-модели сформируется стандартный файл отчета моделирования (СФО), который включает результаты моделирования и информацию об ошибках. Имя СФО то же, что и имя GPSS/H-модели, но с расширением .lis, а не .gps (primer1.lis).

СФО состоит из двух основных частей: программной части (последовательность команд GPSS/H-программы) и результатов моделирования. Результаты моделирования включают разделы времени, блоков и объектов. Ниже приведен фрагмент СФО модели технологии изготовления деталей.

Simulation begins.
RELATIVE CLOCK: 2867.4714 ABSOLUTE CLOCK: 2867.4714

BLOCK	CURRENT	TOTAL	BLOCK	CURRENT	TOTAL
1		90	11		50
2		90	12		50
3		90	13		50
4	2	88	14		50
5		88			
6	2	88			
7		86			
8		86			
9	36	86			
10		50			

--AVG-UTIL-DURING--

FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL

STA2	0.961	50	55.095	AVAIL

STORAGE	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL
STA1	0.951			88	61.949	AVAIL	100.0

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT
LINE1	7	2.454	90	11	12.2	78.192
LINE2	36	16.693	86	1	1.2	556.591

Раздел времени содержит информацию:

RELATIVE CLOCK – относительное время моделирования;

ABSOLUTE CLOCK – абсолютное время моделирования.

Раздел блоков содержит информацию:

BLOCK – имя и номер блока;

CURRENT – число транзактов, находящихся в блоке в момент окончания моделирования;

TOTAL – общее число транзактов, вошедших в блок за период моделирования.

Для прибора обслуживания (Facility) выводится информация:

Facility – имя или номер прибора;

TOTAL TIME – доля времени, в течение которого был занят прибор;

ENTRIES – общее количество транзактов, которые занимали прибор;

AVERAGE TIME/XACT – среднее время обслуживания транзакта прибором;

SEIZING XACT – номер транзакта, который в настоящее время обслуживается прибором;

PREEMPTING XACT – номер транзакта с приоритетным прерыванием.

Для многоканального устройства выводится:

Storage – имя или номер многоканального устройства;

TOTAL TIME – коэффициент использования многоканального устройства;

ENTRIES – общее количество транзактов, которые занимали многоканальное устройство;

AVERAGE TIME/XACT – среднее время обслуживания транзакта каналами многоканального устройства;

SEIZING XACT – номер транзакта, который в настоящее время обслуживается многоканальным устройством;

PREEMPTING XACT – номер транзакта с приоритетным прерыванием.

Столбцы отчета об очереди (QUEUE) в СФО включают:

QUEUE – имя или номер очереди;

MAXIMUM CONTENTS – самый большой размер очереди, которая была в ходе моделирования;

AVERAGE CONTENTS – средний размер очереди;

TOTAL ENTRIES – общее число модулей, которые занимали очередь;

ZERO ENTRIES – число модулей, которые сразу попали на обслуживание;

PERCENT ZEROS – процент модулей, сразу попавших на обслуживание;

AVERAGE TIME/UNIT – среднее время нахождения модуля в очереди, включая модули, которые сразу попали на обслуживание;

$AVERAGE TIME/UNIT – среднее время нахождения модуля в очереди, исключая модули, которые сразу попали на обслуживание;

CURRENT CONTENTS – число модулей, находящихся в очереди в момент окончания моделирования.

На основе информации, приведенной в СФО, можно анализировать результаты моделирования системы.

В соответствии с нашим заданием находим нужные нам результаты:

- продолжительность изготовления 50 деталей (RELATIVE CLOCK = ABSOLUTE CLOCK);

- степень использования группы станков типа А (для STORAGE столбец TOTAL TIME) и типа В (для FACILITY STA2 столбец TOTAL TIME);

- средние значения длин очередей перед станками типа А (QUEUE LINE1) и станком типа В (QUEUE LINE2).

Таким образом, продолжительность изготовления 50 деталей – 2868 мин; степень использования группы станков типа А – 95,1 % , а типа В – 96,1 %; средние значения длин очередей перед станками типа А – 2,5 и станком типа В – 16,7.

4.10. Моделирование при установившемся режиме

Измерение коэффициента загрузки прибора – времени, в течение которого транзакт занимает прибор в рассмотренной выше модели, производится сразу же после начала моделирования. Очевидно, что для оценки эффективности функционирования робототехнического комплекса нужны оценки для стационарного (установившегося) режима; после же начала моделирования некоторое время потребуется для достижения этого стационарного режима модели [5]. Измерения, полученные в начале моделирования, могут дать статистически смещенные оценки; поэтому измерения, полученные в начальный период моделирования, следует отбросить, стереть из памяти ЭВМ. Это выполняется с помощью специального оператора RESET.

Оператор RESET – сбрасывает всю накопленную статистику и таймер относительного времени, но не удаляет транзакты из модели и не изменяет последовательность генераторов случайных чисел.

Примеры оператора RESET:

RESET

вся накопленная статистика будет сброшена, но транзакты удалены не будут;

RESET F(SERVER)

вся статистика будет сброшена, за исключением прибора с именем SERVER.

Если этот оператор стоит после оператора START, который задает начальный интервал моделирования, то тогда состояние модели в момент обнуления системного счетчика будет сохранено, а накопленные статистики будут стерты. После сброса статистик моделирование продолжается в течение времени, задаваемого следующим оператором START. Использование операторов RESET и START показано на примере модели технологии изготовления деталей:

```
SIMULATE                    Начало моделирования
STORAGE  S(STA1),2          Задание станков в группе А
GENERATE      RVEXPO(1,28)  Поступление заготовок
ADVANCE RVNORM(3,32,10)     Транспортировка заготовки
QUEUE         LINE1         Занятие очереди перед станками
                            группы А
ENTER         STA1          Занятие станка из группы А
DEPART   LINE1              Освобождение очереди перед станками
                            группы А
ADVANCE 63,9                Обработка на станке группы А
LEAVE         STA1          Освобождение станка группы А
ADVANCE RVNORM(5,12,5)      Транспортировка детали
QUEUE         LINE2         Занятие очереди перед станком типа Б
SEIZE         STA2          Занятие станка типа Б
DEPART   LINE2              Освобождение очереди перед станком
                            типа Б
ADVANCE 55,5                Обработка на станке типа Б
RELEASE  STA2               Освобождение станка типа Б
TERMINATE     1             Поступление деталей в накопитель
START         100           Прогон модели
RESET                       Сброс статистики
START         50            Повторный прогон
END                         Окончание моделирования
```

В модели длина начальной реализации задается первым оператором START, после обслуживания 50 транзактов произойдет сброс накопленных статистик. В СФО нас будут интересовать результаты обработки измерений коэффициента загрузки приборов многоканального устройства с именем STA1 и прибора STA2 и времени занятия этого прибора транзактами, полученными при пропуске 100 транзактов через модель, что задает второй оператор START, входящий в состав модели. СФО может быть получен после прогона модели способом, описанным выше.

Вопросы выбора длины реализации статистической модели в настоящем учебном пособии не рассматриваются, они достаточно полно отражены в изданиях [1, 2].

При запуске приведенной выше GPSS/H-программы на экране появится сообщение об ошибке:

ERROR NUMBER 411 – Out of COMMON: Add/change REALLOCATE Stmt? See release notes

(ОШИБКА 411 – «Превышение COMMON» – Добавить/изменить REALLOCATE?).

Эта ошибка возникает потому, что при прогоне модели используется так называемая динамическая область памяти – COMMON. По умолчанию студенческая версия GPSS/H обращается к 10 000 байтам памяти COMMON. Некоторым программам для успешного прогона необходимо больше чем 10 000 байт области памяти COMMON. Если такая необходимость существует, вы можете использовать специальную опцию MAXCOM (максимизировать COMMON) или команду REALLOCATE (перераспределить). Использование MAXCOM или REALLOCATE зависит только от вас. Если совместно используются MAXCOM и REALLOCATE, то MAXCOM отменяет REALLOCATE.

Опция MAXCOM используется в командной строке, при запуске GPSS/H-программы. Например,

GPSSH PRIMER MAXCOM

MAXCOM заставит GPSS/H максимально зарезервировать область памяти COMMON, до 32 720 байтов для студенческой версии. Фактическое значение области памяти COMMON, которое MAXCOM будет способен зарезервировать, зависит только от сложности вашей программы. Это не зависит от конфигурации ЭВМ или программного обеспечения.

Команда REALLOCATE используется непосредственно в программе, чтобы увеличить область памяти COMMON. REALLOCATE требует определения точного объема памяти. Синтаксис REALLOCATE :

REALLOCATE COM, bytes

«COM» – это A-операнд, который является для GPSS/H кодом области памяти COMMON. B-операнд определяет число байтов памяти, которые необходимо добавить в область памяти COMMON.

REALLOCATE удобнее размещать в начале GPSS/H-программы. Если при помощи REALLOCATE задано меньшее значение области памяти COMMON, которое необходимо для прогона программы, то появится сообщение с информацией о том, какого размера должна быть область памяти COMMON. Это позволяет быстро исправить ошибку.

Например,

REALLOCATE COM,20000

Значение области памяти COMMON удвоится по сравнению со значением, установленным по умолчанию. Вообще, если

GPSS/H-программа не выполняется при 10 000 байтов области памяти COMMON, то она должна заработать при значении в два раза большем. Следовательно, чтобы осуществить успешный прогон рассмотренного в этом параграфе примера моделирования, к модели необходимо добавить команду REALLOCATE:

```
SIMULATE                          Начало моделирования
STORAGE       S(STA1),2           Задание станков в группе А
REALLOCATE    COM,20000           Увеличение размера динамиче
                                  ской памяти
GENERATE      RVEXPO(1,28)        Поступление заготовок
ADVANCE       RVNORM(3,32,10)     Транспортировка заготовки
.
.
.
END
```

4.11. Изменение последовательности псевдослучайных чисел

Для розыгрыша случайных чисел в GPSS/H используются 8 датчиков равномерного распределения в интервале (0, 1). Эти датчики являются датчиками псевдослучайных чисел, т.е. если каждый цикл работы генератора псевдослучайных чисел начинается с одними и теми же исходными данными (начальными значениями), то на выходе получаются одинаковые последовательности чисел. Следовательно, при многократных прогонах модели стохастического процесса датчики будут при обращении к ним выдавать одну и ту же последовательность чисел. Т.е. результаты нескольких прогонов будут одинаковыми (не случайными). Для достоверности выводов на основе результатов моделирования необходимо провести несколько прогонов (действительно случайных) и найти среднее значение определяемой величины. Для этого необходимо изменить последовательность (алгоритм) случайных чисел. В GPSS/H это возможно сделать, используя оператор RMULT.

Оператор RMULT – устанавливает начальные значения датчиков псевдослучайных чисел и изменяет их последовательность. По умолчанию начальные значения всех датчиков разные.

В операндах A, B, C, D, E, F, G и H записываются начальные значения для датчиков 1-го, ..., 8-го соответственно. Эти значения могут быть любыми, но не превышающими 10 знаков.

Например,

RMULT 123,15,,,347

начальные значения (и последовательность псевдослучайных чисел) 1-, 2- и 5-го датчиков изменяются. Начальные значения всех остальных датчиков остаются неизменными.

Оператор RMULT ставят:

а) до первого оператора START для установления начальных значений генераторов;

б) между операторами START для восстановления начальных значений генераторов;

в) между операторами START для определения новых начальных значений генераторов.

При прогоне моделей в рассмотренных выше примерах можно заметить, что при использовании различных датчиков случайных чисел результаты тоже различны. Возникает вопрос: «А какой из этих результатов верный?». Для получения такого результата необходимо провести несколько прогонов, изменяя последовательность псевдослучайных чисел (изменяя номера генераторов), а затем взять среднее значение либо провести достаточно долгое моделирование.

4.12. Проведение нескольких экспериментов за один прогон модели

При проведении имитационных экспериментов часто приходится выполнять несколько прогонов, лишь слегка изменяя модель. Например, необходимо исследовать влияние изменения интервала движения какого-либо транспортного средства или интервалов потока заготовок на показатели всей системы, т.е. провести эксперименты на модели с различным временем задержки транзакта в блоке ADVANCE. Для этого нужно запустить модель, проанализировать информацию в сформированном стандартном файле отчета моделирования, исправить временную задержку в нужном блоке модели, опять запустить модель и повторить эту последовательность действий.

Было бы удобно, если бы вместо серии выполнений независимых заданий с изменением программы все это можно было сделать за один прогон модели, а именно:

1. После завершения моделирования и распечатки статистических данных внести некоторые изменения в модель:

а) переопределить операнды одного или нескольких блоков модели без изменения типов самих блоков;

б) переопределить типы блоков;

в) расширить модель за счет добавления сегментов.

2. После модификации модели подготовить ее к повторному прогону:

а) сбросить всю статистику модели в нулевое состояние;

б) удалить все транзакты, которые оставались в модели в конце моделирования предыдущего прогона.

Каждый из перечисленных шагов может быть выполнен средствами языка GPSS/H. Шаг 2 выполняется при помощи оператора CLEAR.

Оператор CLEAR – удаляет все транзакты из модели, сбрасывает таймер абсолютного и относительного времени, а также всю статистику относительно объектов (если нет ограничений). Не воздействует на переменные и на генераторы случайных чисел.

Для изменения значений переменных используют оператор LET. Изменить последовательность значений, произведенных генератором случайных чисел, возможно при помощи оператора RMULT. Эти операторы будут рассмотрены ниже.

Формат CLEAR:

CLEAR n1,n2,...

необязательные операнды n1, n2, ... – имена объектов, параметры которых не должны сбрасываться (по умолчанию вся статистика сбрасывается).

Если CLEAR вставлен между двумя START, то после первого прогона все транзакты в модели будут удалены и вся статистика сброшена. GPSS/H продолжит выполнение модели, когда встретится со вторым оператором START:

```
START  1
CLEAR
START  1
```

Пример моделирования

Допустим, что в нижеприведенной модели необходимо определить загрузку прибора при различных временах обслуживания (13.4; 11.5; 9.2).

```
SIMULATE
GENERATE      19,7      Поступление заявок
SEIZE         NIC       Занятие прибора
ADVANCE       13,4      Обслуживание заявки
RELEASE       NIC       Освобождение прибора
TERMINATE     1         Удаление заявки
START         100       Запуск модели
END
```

Можно выполнять несколько серий последовательных прогонов, меняя в модели временную задержку в блоке ADVANCE. При использовании CLEAR имеется возможность объединить эти три прогона в один этап моделирования. Измененная модель, позволяющая это сделать, приведена ниже:

```
SIMULATE
GENERATE      19,7      Поступление заявок
SEIZE         NIC       Занятие прибора
BOX  ADVANCE  13,4      Обслуживание заявки
RELEASE       NIC       Освобождение прибора
TERMINATE     1         Удаление заявки
START         100       Запуск модели
CLEAR                   Обнуление статистики
BOX  ADVANCE  11,5      Переопределение операндов блока
START         100       Запуск модели
CLEAR                   Обнуление статистики
BOX  ADVANCE  9,2       Переопределение операндов блока
START         100       Запуск модели
END
```

При трансляции модели может появиться предупреждение:

THE FOLLOWING ENTITIES HAVE BEEN MULTIPLY DEFINED, означающее, что нескольким блокам присвоено одно имя. Однако интерпретатор не считает это за ошибку. Это предупреждение о том, что разработчик может ненамеренно присвоить одно и то же имя разным блокам.

4.13. Моделирование непоследовательных операций

Во всех приведенных выше моделях технологических процессов транзакты переходят последовательно от блока к блоку, т.е. отображают последовательные операции, происходящие в системе. Если при моделировании возникает необходимость нарушить такую последовательность, то в модели надо нарушить последовательность движения транзактов. Для этого используют блок TRANSFER.

Блок TRANSFER (передать) – перенаправляет транзакты в блок, отличный от последующего (рис. 24).

Блок может использоваться в 8 режимах. Рассмотрим 3 основных режима.

Режим безусловной передачи

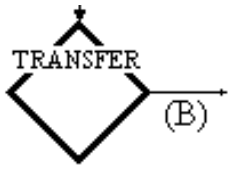


Рис. 24. Блок TRANSFER в режиме безусловной передачи

В операнде A ставится запятая (,). В операнде B записывается имя блока, в который должен перейти транзакт.

Пример блока TRANSFER в режиме безусловной передачи:

```
TRANSFER     ,BOX
```

Когда транзакты входят в блок, они сразу же пытаются войти в блок BOX. Если последний отказывает в этом, транзакты остаются в блоке TRANSFER.

Ряд сложных систем, и в том числе производственных, сводятся к замкнутым системам массового обслуживания. В этих СМО общее число заявок в течение всего интервала моделирования системы остается постоянным – заявки не покидают СМО, а циркулируют в ней, последовательно изменяя свои состояния в моменты перехода от одной фазы обслуживания к другой. Например, транспортная система некоторого производства, содержащая N транспортных средств или сборочное производство, в котором участвуют N человек.

При построении GPSS-моделей подобных систем число транзактов в модели должно оставаться постоянным в течение всего интервала моделирования. Это может быть достигнуто благодаря использованию блока GENERATE с операндом D, который выдает в модель заданное число транзактов, и блока TRANSFER в режиме безусловной передачи, помещенного в конце модели и возвращающего транзакты в начало модели.

Пример моделирования

Рассмотрим пример GPSS/H-модели сборочного производства [4]. Производство изделий включает процесс сборки и обжиг в печи. Содержание печи обходится дорого, поэтому 4 сборщика используют одну печь, в которой одновременно можно обжигать только одну деталь. Сборщик не может начать новую сборку, пока в печи находится предыдущая деталь. Каждый сборщик выполняет следующие действия: сборку следующей детали; ожидание возможности использования печи; использование печи; возврат к сборке.

Время, необходимое на сборку, колеблется в интервале от 25 до 35 минут (распределение равномерное), а процесс обжига распределен нормально со средним значением 8 минут и среднеквадратическим отклонением 2 минуты. Необходимо промоделировать данное производство в течение 8-часового рабочего дня.

Пусть транзакты отображают сборщиков, приход которых смоделируем блоком GENERATE с операндом D, равным 4. За единицу модельного времени примем 1 минуту. Тогда время сборки отобразим блоком ADVANCE со средним 30 и размахом 5. Работу печи смоделируем прибором с именем PECH. После того как транзакт завершает использование прибора, моделирующего печь, он должен быть возвращен назад. Отобразим это условие при помощи блока TRANSFER в режиме безусловной передачи, который перенаправляет транзакты в блок следующей сборки. Для организации моделирования в течение 480 минут используем сегмент, содержащий блок GENERATE с соответствующим операндом A.

Блочная интерпретация модели сборочного производства представлена на рис. 25.

Созданная на основе блок-схемы GPSS/H-программа представлена ниже:

```
        SIMULATE                    Начало моделирования
        GENERATE    ,,,4            Приход сборщиков
BACK    ADVANCE     30,5            Сборка
        SEIZE       PECH            Занятие печи
        ADVANCE     RVNORM(2,8,2)   Обжиг
        RELEASE     PECH            Освобождение печи
        TRANSFER    ,BACK           Переход к сборке
        GENERATE    480             Выход транзакта-таймера
        TERMINATE   1               Удаление транзакта-таймера
        START       1               Запуск модели
        END                         Конец моделирования
```

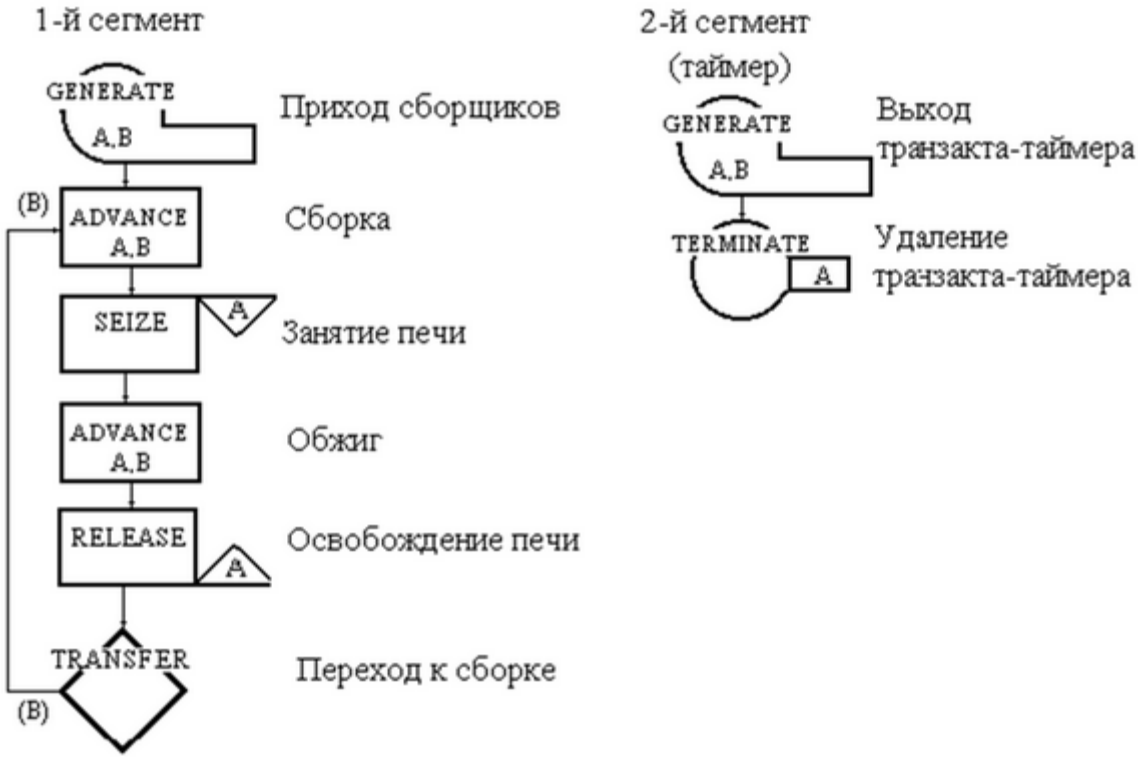


Рис. 25. Блочная интерпретация GPSS/H-модели процесса сборки

Режим статистической передачи

Часто при моделировании технологических процессов необходимо перенаправлять транзакты случайным образом в два различных блока модели. Такие ситуации можно смоделировать блоком TRANSFER в режиме статистической передачи (рис. 26).

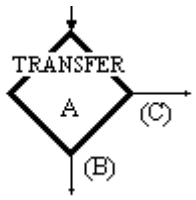


Рис. 26. Блок TRANSFER
в режиме статистической передачи

В операнде A записывают частоту передачи транзактов в блок C (это число должно быть больше 0 и меньше 1 и всегда начинаться с десятичной точки). В операнде B записывают имя блока, в который переходят транзакты с частотой равной 1 (частота в операнде A). Розыгрыш направления передачи транзакта производится с помощью датчиков равномерно распределенных случайных величин в интервале (0, 1), входящих в GPSS/H.

Примеры блока TRANSFER в режиме статистической передачи:

TRANSFER .25,BOX,PLAY

транзакты, входящие в блок, в 25 % случаев будут передаваться в блок с именем PLAY. В остальных 75 % случаев они будут переданы в блок с именем BOX.

TRANSFER .33,,FAX

транзакты, входящие в блок, будут переданы в 33 % случаев в блок с именем FAX. В остальных случаях они попадут в следующий по порядку блок.

Пример моделирования

Приведем пример модели гибкой производственной системы, содержащей два робототехнических комплекса, где блок TRANSFER используется в режиме статистической и безусловной передачи.

Пусть ГПС содержит два независимо функционирующих робототехнических комплекса (РТК). В ГПС поступает равномерный поток заданий на обработку. Для выполнения одних заданий требуется РТК1, для выполнения других – РТК2. Выбор РТК происходит случайным образом: с вероятностью, равной 0.35, задание направляется на РТК1 и с вероятностью 0.65 (1–0.35) – на РТК2. Для упрощения времена между поступлениями заданий в ГПС, времена работы РТК по выполнению заданий примем равномерно распреде-

ленными со средними значениями 100, 80, 200 и размахом 40, 50, 20 соответственно. Требуется построить модель и определить загрузку обоих РТК и среднее количество скопившихся перед ними заданий.

Графическая интерпретация модели представлена на рис. 27.

GPSS/H-программа приведена ниже:

```
      SIMULATE
      GENERATE     100,40          Поступление заданий в ГПС
      TRANSFER     .65,OH1,OH2     65 % идет в РТК2, а 35 % в РТК1
OH1   QUEUE        1               Ожидание освобождения РТК1
      SEIZE        RTK1            Занятие РТК11
      DEPART       1               Конец ожидания
      ADVANCE      80,50           Обработка в РТК1
      RELEASE      RTK1            Освобождение РТК1
      TRANSFER     ,FIN            Задание выполнено
OH2   QUEUE        2               Ожидание освобождения РТК2
      SEIZE        RTK2            Занятие РТК2
      DEPART       2               Конец ожидания
      ADVANCE      200,20          Обработка в РТК2
      RELEASE      RTK2            Освобождение РТК2
FIN   TERMINATE                    Задание выполнено
      GENERATE     &TIM            Моделирование 24 часов
      TERMINATE    1
      START        1
      END
```

В модели использованы два прибора и две очереди. Прибор с именем RTK1 использован для моделирования работы РТК1, прибор с именем RTK2 – для моделирования работы РТК2. Очередь 1 служит для сбора информации о времени нахождения на ГПС заданий, направляемых на РТК1, очередь 2 – времени нахождения в ГПС заданий, направляемых на РТК2. Поток транзактов создается блоком GENERATE, после чего следует вероятностное прореживание потока транзактов с помощью блока TRANSFER. Этот блок с вероятностью 0.35 направляет транзакты в блок с именем OH1 и с вероятностью 0.65 в блок с именем OH2. Блок TRANSFER безусловного типа в модели использован для передачи транзактов, вышедших из прибора RTK1, в блок TERMINATE с именем FIN.

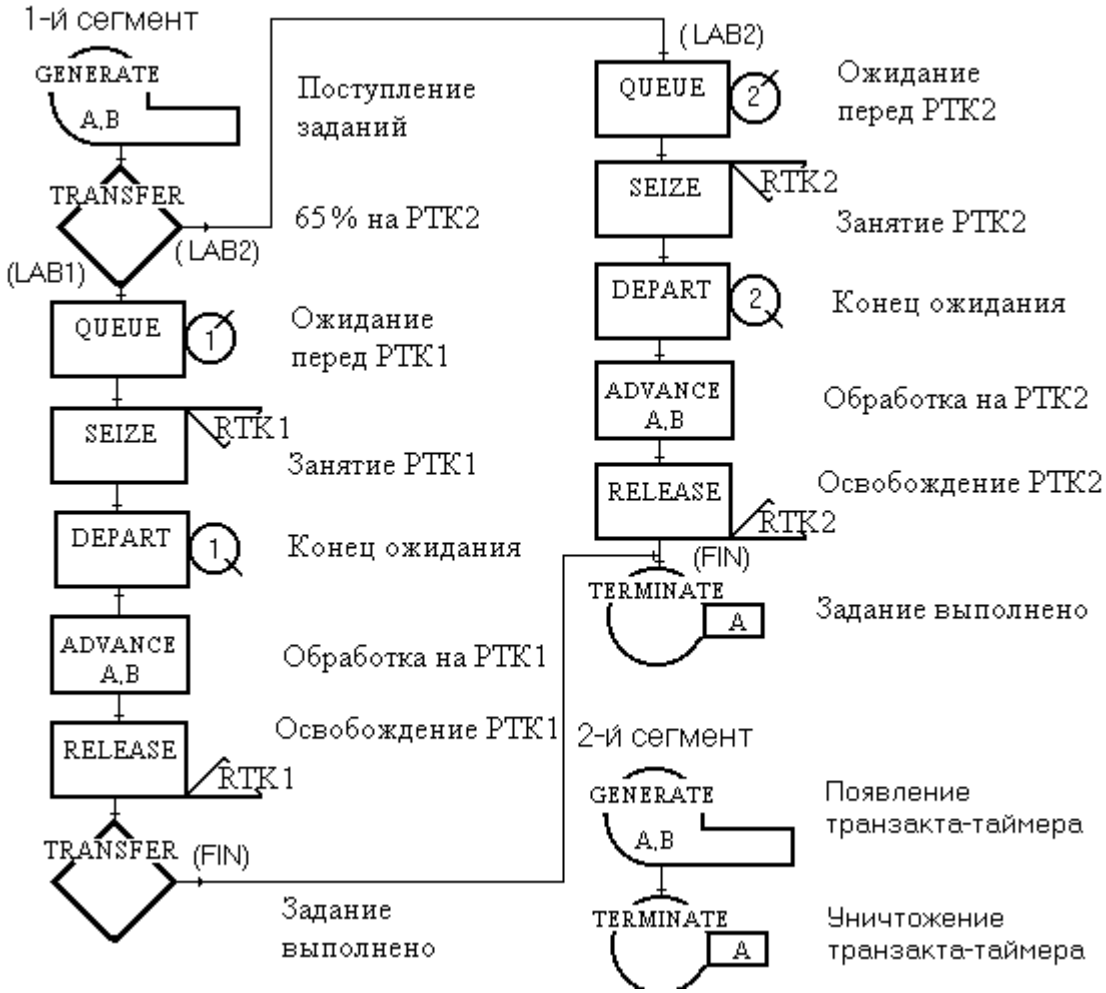


Рис. 27. Блок-схема GPSS/H-модели ГПС

Режим BOTH

Когда требуется передать транзакты не случайным образом в один из двух заданных блоков, используют блок TRANSFER в режиме BOTH (рис. 28).

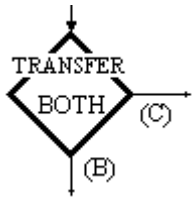


Рис. 28. Блок TRANSFER в режиме BOTH

В операнде A записывается слово BOTH. В операнде B записывается имя блока, в который будет осуществляться начальная попытка передачи транзакта. Если она неудачна (например, занят блок

SEIZE), то транзакт будет передан в блок с именем, записанным в операнде C. Если и этот блок откажет во входе, то транзакт останется в блоке TRANSFER, пока блоки, указанные в операндах B или C, не освободятся. Первым проверяется блок с именем в операнде B.

Примеры блока TRANSFER в режиме BOTH:

TRANSFER BOTH,BOX,PLAY

транзакт будет пытаться войти в блок с именем BOX, если эта попытка неудачна, то он будет пытаться войти в блок с именем PLAY. Если и эта попытка неудачна, транзакт останется в блоке TRANSFER, пока не освободится блок BOX или блок PLAY.

TRANSFER BOTH,,COM

транзакт будет пытаться войти в следующий по порядку блок, если эта попытка неудачна, то он будет пытаться войти в блок с именем COM.

4.14. Стандартные числовые атрибуты

В процессе моделирования интерпретатор автоматически регистрирует и корректирует некоторую информацию, касающуюся различных элементов, используемых в модели. Такой информацией являются текущие результаты моделирования: счетчики блоков, загрузки приборов и многоканальных устройств, средние времена пребывания в очередях и т.д. Эти результаты могут быть использованы в процессе моделирования. Более того, процесс моделирования может управляться динамически в зависимости от их значений. Например, интенсивность, с которой прибор обслуживает заявки, может зависеть от числа заявок, ожидающих обслуживания. При возрастании длины очереди прибор может работать быстрее.

Для оперирования с текущими результатами в GPSS/H используются стандартные числовые атрибуты (СЧА). Основные СЧА приведены в табл. 6 – 11.

Например, в N(COMP) записывается общее число вошедших в блок с именем COMP транзактов; в FR(SERVER) записывается коэффициент использования прибора под именем SERVER.

Таблица 6

СЧА времени

СЧА	Описание
AC1 C1	Время, прошедшее, начиная с начала моделирования (абсолютное время) Время, отсчитанное после того, как GPSS/H встретил оператор RESET (относительное время). Если оператора RESET нет в модели, то AC1 = C1.

Таблица 7

СЧА блоков

СЧА	Описание
W(n) N(n)	Число транзактов в текущий момент времени в блоке с именем или номером n Число транзактов, вошедших в блок, с именем или номером n за все время моделирования

Таблица 8

СЧА транзактов

СЧА	Значение
PB(n), PH(n), PF(n), PL(n)	Номер или имя соответственно байтового, полусловного, полнословного и действительного параметров транзакта
PR	Уровень приоритета транзакта
XID1	Номер обрабатываемого в данный момент времени транзакта

Таблица 9

СЧА приборов

СЧА	Значение
F(n)	Занятость прибора; F(n) = 0, если прибор свободен, и F(n) = 1, если прибор занят
FT(n)	Время обслуживания транзакта прибором с именем n
FR(n)	Коэффициент использования прибора с именем n

Таблица 10

СЧА многоканальных устройств (МУ)

СЧА	Значение
R(n)	Свободная емкость МУ под именем n
S(n)	Текущее содержимое МУ под именем n
SA(n)	Среднее содержимое МУ под именем n
SC(n)	Число входов в МУ
SE(n)	1, если МУ не заполнено в настоящий момент, 0, если заполнено
SF(n)	1, если МУ заполнено в настоящий момент, 0, если не заполнено
SM(n)	Максимальное содержимое МУ под именем n
SR(n)	Коэффициент использования МУ под именем n

Например,

R(BUFFER)

в СЧА записывается число модулей МУ BUFFER, которые в настоящее время не используются.

SF(5)

в СЧА SF(5) принимает значение 1, если МУ под номером 5 в настоящее время заполнено; в противном случае МУ(5) принимает значение 0 (используется в TEST и в операторе FUNCTION).

Таблица 11

СЧА очереди

СЧА	Значение
Q(n)	Текущее содержимое очереди под именем n
QA(n)	Средний размер очереди n
QC(n)	Число входов в очередь n
QM(n)	Максимальное содержимое очереди n
QT(n)	Среднее время пребывания транзакта (из расчета QC)
QX(n)	Среднее время пребывания транзакта (из расчета QZ)
QZ(n)	Число нулевых входов (без задержки в очереди)

Например,

Q (LINE)

в СЧА записывается текущее число транзактов в очереди под именем LINE.

Все перечисленные СЧА могут использоваться в качестве операндов блоков и аргументов функций.

4.14.1. Атрибуты транзактов

Атрибуты транзактов – характеристики, связанные с транзактами. Они есть у каждого транзакта и изменяются при движении транзактов по модели. Атрибуты транзактов делятся на две категории: встроенные (номер транзакта – XID1; номер блока, в котором находится транзакт; номер блока, куда войдет транзакт; время нахождения транзакта в модели – M1; уровень приоритета транзакта – PR) и определяемые пользователем или параметры (байтовый, полусловный, полнословный, действительный).

В GPSS/H возможно изменять только два встроенных атрибута: метку времени (присваивается транзакту, как только он создается блоком GENERATE) и уровень приоритета транзакта (первоначально задается в блоке GENERATE). В процессе перемещения транзактов по модели их параметры могут устанавливаться и модифицироваться в соответствии с заданной пользователем логикой. Эти значения можно использовать в качестве операндов блоков или аргументов функций.

Имя параметра состоит из двух частей: группового имени и номера конкретного члена этой группы. Групповым именем являются символы PH, PF, PB, PL в зависимости от типа параметра (полусловного, полнословного, байтового и действительного соответственно).

Тип и количество параметров каждого типа для транзакта определяются посредством блока GENERATE в операндах F, G, H (значением по умолчанию является PH с количеством параметров равным 12).

Пример задания параметров транзактов:

GENERATE 7,,,,,4PF,8PL

транзакты будут создаваться каждые 7 единиц времени. Каждый транзакт будет иметь 4 полнословных параметра и 8 действительных.

GENERATE 5

транзакты будут создаваться каждые 5 единиц времени. Каждый транзакт будет иметь 12 полусловных параметров.

Полусловный параметр (PH) может быть целым значением в интервале от –32,768 до 32,767. Полнословный параметр (PF) может быть целочисленным значением в интервале от –2 147 483 648 до 2 147 483 647. Байтовый параметр (PB) может быть целочисленным значением в интервале от –128 до 127. Действительный параметр (PL) может иметь значения с плавающей запятой в интервале от – 2 147 483 648 до 2 147 483 647.

При входе транзакта в модель начальным значением всех его параметров является 0. Значения параметров определяет пользователь. Чаще всего это делают путем присвоения им некоторых числовых значений в соответствии со схемой кодирования. После этого значения параметров можно использовать явным образом.

Для примера рассмотрим гибкую производственную систему, в которой разделяются детали в зависимости от их веса и вида прошедшей обработки. Пусть транзакт – это деталь. Характеристики каждой детали могут быть заданы в параметрах в соответствии со схемой кодирования, приведенной в табл. 12.

Таблица 12

Вариант интерпретации значений параметров транзактов

Значение PB3	Интерпретация (вид обработки)	Значение PF7	Интерпретация (вес, кг)
1	Токарная	1	10
2	Фрезерная	2	20
3	Сверлильная	3	30

Например, если транзакт имеет в качестве параметров PB3 и PF7 значения 3 и 1 соответственно, это означает – просверленная деталь весом 10 кг.

Значения параметров транзактов можно использовать в качестве операндов блоков или аргументов функций.

Примеры использования параметров транзактов:

ADVANCE PF9

транзакт, попавший в блок, будет задержан на значение времени, которое записано в 9-м полнословном параметре этого транзакта.

TRANSFER ,PH5

транзакт будет послан к блоку, номер которого определен в 5-м полусловном параметре транзакта.

4.15. Проверка числовых выражений

Сравнить числовые выражения или соотношение между двумя стандартными числовыми атрибутами можно при помощи блока TEST.

Блок TEST (проверить) – предназначен для сравнения числовых выражений, а также СЧА и перенаправления транзактов (рис. 29).

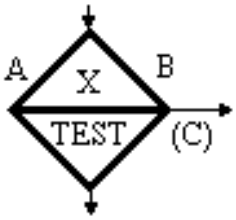


Рис. 29. Блок TEST

Значения операндов блока представлены в табл. 13.

Примеры блока TEST:

TEST E Q1,Q2

транзакт будет задержан в предыдущем блоке до тех пор, пока содержимое очереди 1 не станет равно содержимому очереди 2.

TEST LE S(BUFFER),4,EXIT

если текущее содержимое многоканального устройства BUFFER меньше или равно 4, то транзакт переходит в следующий блок, в противном случае транзакт переходит в блок с именем EXIT.

Таблица 13

Значения операндов блока TEST

Операнд	Значение	Результат по умолчанию
A	Имя первого СЧА	Ошибка
B	Имя второго СЧА	Ошибка
X	Оператор отношения, используемый при проверке	Ошибка

	Оператор отношения	Вопрос, подразумеваемый в блоке TEST	
	G GE E NE LE L	A > B A ≥ B A = B A ≠ B A <= B A < B	
C	Необязательный; имя блока, в который переходит проверяющий транзакт, если ответ на вопрос, подразумеваемый оператором отношения, отрицателен		Проверка производится в режиме отказа

4.16. Присвоение числовых значений параметрам транзакта

По умолчанию значения всех параметров транзактов равны 0. Для записи значений в параметры в GPSS/H используется блок ASSIGN.

Блок ASSIGN (назначить) – присваивает значения параметрам транзактов (рис. 30).

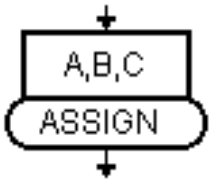


Рис. 30. Блок ASSIGN

В операнде A записывают номер параметра, в операнде B – число или выражение, в операнде C – тип параметра транзакта.

Примеры блока ASSIGN:

ASSIGN 3,25,PF

3-му полнословному параметру транзакта присваивается значение, равное 25.

ASSIGN 1,7*Q(OTCH),PH

значение текущего содержимого очереди OTCH умножается на 7 и присваивается первому полусловному параметру транзакта.

ASSIGN 5,FR(COMP),PL

коэффициент использования прибора COMP записывается в 5-й действительный параметр транзакта.

4.17. Изменение приоритета транзакта

Транзакты в GPSS/H-моделях имеют атрибут, называемый приоритетом (PR). Изначально приоритет транзактов назначается в блоке GENERATE в E-операнде (значение по умолчанию – 0). Если транзакты ожидают обслуживания (находятся в очереди) и у всех транзактов равный приоритет, то транзакты обслуживаются по принципу «первым пришел – первым обслужен». В ранее приведенных примерах все транзакты имели нулевой приоритет. Часто этого не достаточно для моделирования реальных систем. Некоторые транзакты могут иметь более высокий приоритет, чем другие. Например, заготовки, требующие меньшего времени обработки, могут идти на обработку раньше заготовок с большим временем обработки. Для этого требуется изменить значение уровня приоритета транзакта.

Блок PRIORITY (назначить приоритет) – изменяет уровень приоритета транзакта (рис. 31).

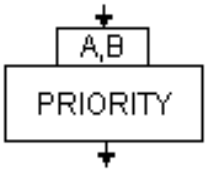


Рис. 31. Блок PRIORITY

В операнде A записывают новый уровень приоритета транзакта.

Пример блока PRIORITY:

PRIORITY 1

уровень приоритета транзакта устанавливается равным 1.

4.18. Пример компьютерной имитации

Разработку модели компьютерной имитации покажем на примере отображения работы некоторого производства [4].

В цехе 50 станков разных типов (20 станков типа A и 30 типа B) работают по 8 часов в день и по 5 дней в неделю. Имеется резерв

арендуемых станков – 1 типа A и 2 типа B. В любой момент времени любой станок может выйти из строя. В этом случае его заменяют резервным. Сломанный станок отправляют в ремонтную мастерскую, где его чинят и возвращают в цех, но уже в качестве резервного (рис. 32). Починку станков в ремонтной мастерской осуществляют три человека, при этом на ремонт станка типа A уходит в среднем 7±3 часа, а на ремонт станка типа B – 5±2 часа (распределение равномерное). Время наработки на отказ станка составляет 157±25 часов (распределение равномерное).

Необходимо промоделировать производство в течение 1 года (без выходных и праздников; 8-часовой рабочий день) и определить количество выходивших из строя станков по типам и коэффициент загрузки ремонтников.

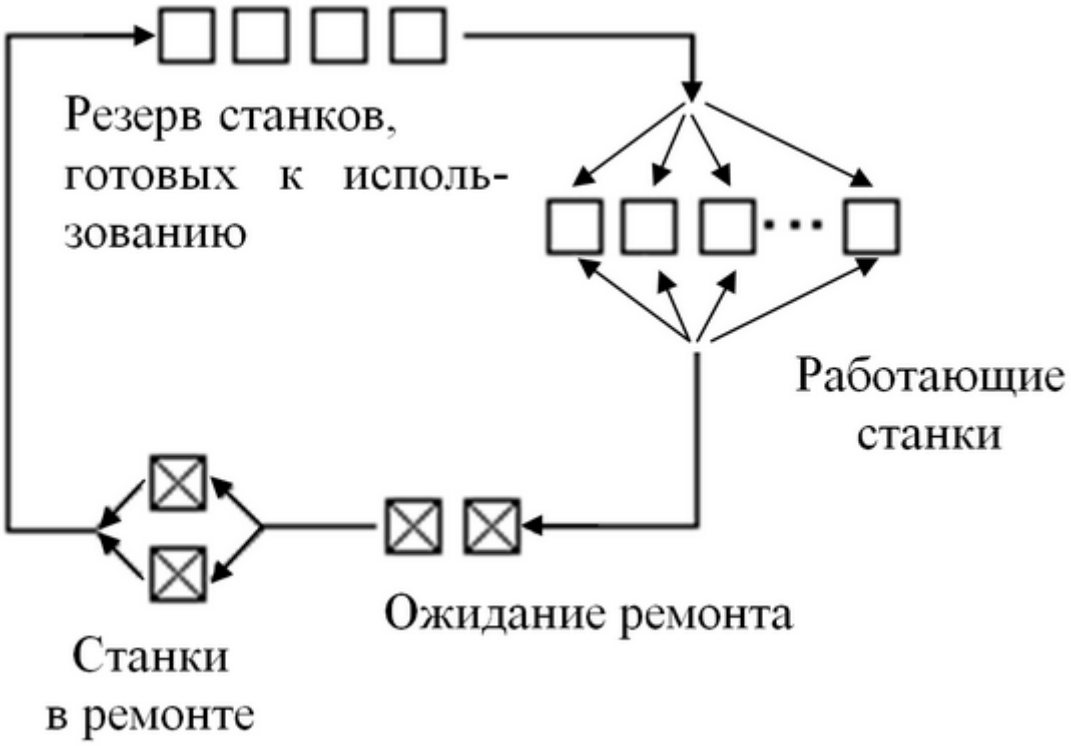


Рис. 32. Схема производства

Блок схема GPSS/H-модели данного производства представлена на рис. 33.

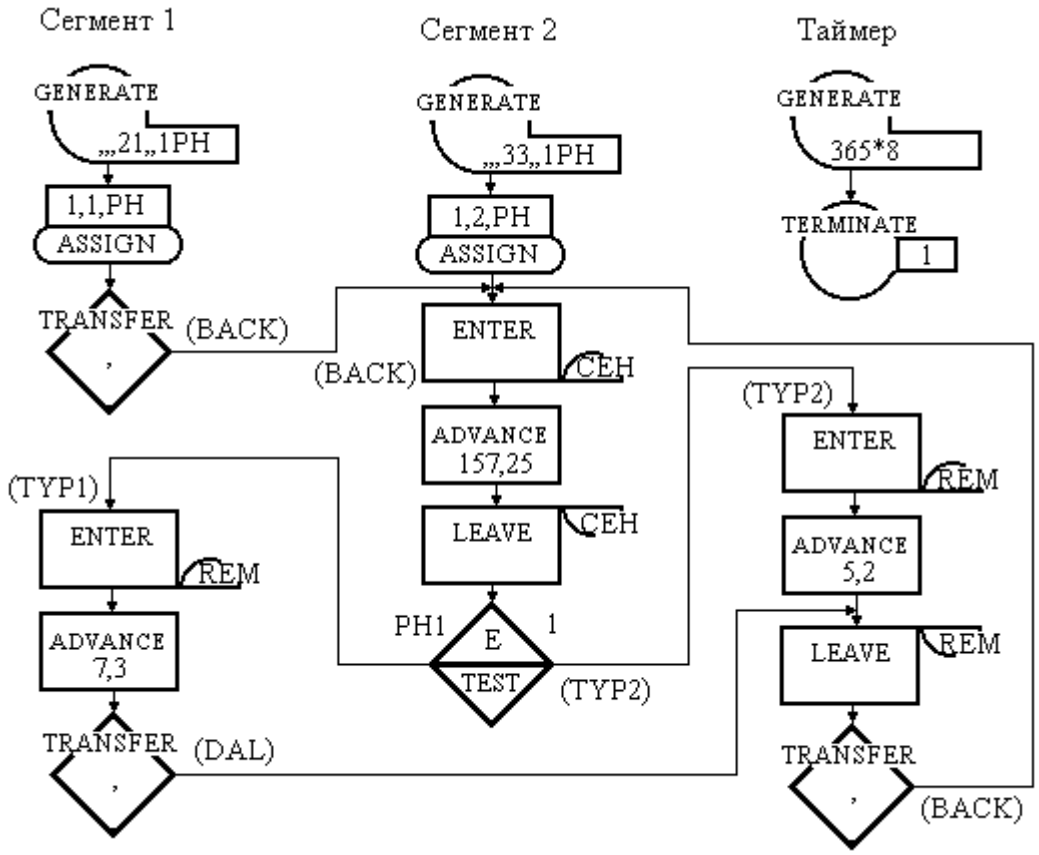


Рис. 33. Блок-схема GPSS/H-модели производства

Разработанная в соответствии с блок-схемой GPSS/H-программа приведена ниже:

```
            SIMULATE
            STORAGE     S(REM),3/S(CEH),50   Кол-во ремонтников
                                             и станков
            GENERATE    ,,,21,,1PH   Поступление станков типа A
            ASSIGN      1,1,PH       Отметка станков типа A
            TRANSFER    ,BACK        Передача станков типа A в цех
            GENERATE    ,,,33,,1PH   Поступление станков типа B
            ASSIGN      1,2,PH       Отметка станков типа B
BACK        ENTER       CEH          Начало работы станков обоих
                                     типов
            ADVANCE     157,25       Наработка на отказ
            LEAVE       CEH          Поломка станка
            TEST E      PH1,1,TYP2   Передача станка на ремонт
                                     в зависимости от типа
TYP1        ENTER       REM          Занятие ремонтников
            ADVANCE     7,3          Ремонт станка типа A
            TRANSFER    ,DAL         Освобождение ремонтников
```

```
TYP2    ENTER     REM     Занятие ремонтников
        ADVANCE   5,2     Ремонт станка типа B
DAL     LEAVE     REM     Освобождение ремонтников
        TRANSFER  ,BACK   Передача станков в цех
        GENERATE  365*8   Моделирование 1 года работы в
                          часах
        TERMINATE 1
        START     1
        END
```

После запуска модели сформируется стандартный файл отчета моделирования. Часть этого файла представлена ниже (интересующие нас результаты подчеркнуты).

```
Simulation begins.
RELATIVE CLOCK: 2920.0000   ABSOLUTE CLOCK: 2920.0000

BLOCK CURRENT    TOTAL  BLOCK CURRENT    TOTAL
1                       21  11           2    345
2                       21  12                343
3                       21  TYP2              556
4                       33  14                556
5                       33  DAL                    899
BACK                    951 16                2    899
7             50        951 17                     1
8                       901 18                     1
9                       901
TYP1                    345

        --AVG-UTIL-DURING--
 STORAGE  TOTAL  AVAIL  UNAVL    ENTRIES    AVERAGE
                 TIME    TIME     TIME               TIME/UNIT
   REM    0.602                              901        5.853
   CEH    0.990                              951      152.059
```

Таким образом, за время моделирования из строя выходили 345 станков типа A и 556 типа B. Коэффициент загрузки ремонтников составил 0.602.

5. КОМПЬЮТЕРНАЯ АНИМАЦИЯ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

5.1. Введение в язык компьютерной анимации Proof Animation

Анимация – это отображение динамики работы оборудования на компьютерной мнемосхеме технологического процесса, формируемое имитационной моделью процесса.

Язык компьютерной анимации Proof Animation может работать в комплексе с универсальными BASIC, C++, FORTRAN, PASCAL или специализированными GPSS, SIMAN, SIMPLE, SIMSCRIPT, SLAM языками, создающими стандартные ASCII-файлы. Proof Animation является универсальным программным обеспечением системы анимации для персональных компьютеров, позволяющим проектировщику создавать любые цветные двухмерные и изометрические рисунки, формировать набор команд для движения элементов рисунка, редактировать последовательность движений.

Применение компьютерной анимации дает возможности:

- проверить адекватность модели объекту в деталях и в целом;
- выявить трудноуловимые ошибки имитационного моделирования;
- доказать заказчику правильность работы имитационной модели;
- проиграть различные ситуации для непрограммирующего пользователя;
- вести обучение на модели.

Простые инструкции языка Proof Animation сводятся к заданию времени и конечных точек перемещения нарисованных объектов. Изображение процесса строят из статических и динамических элементов. Для процессов машиностроительных производств статическими элементами являются контуры участка цеха, станки, накопители и т.д. Их рисуют с помощью линий, дуг, прямоугольников, окружностей, имеющихся в меню программы. Возможны выбор цвета, ширины и типа линии; просмотр размеров и ориентации объекта до его ввода в рисунок; изображение кривой любой формы путем передвижения «мыши». Статическими элементами могут быть пояснительные надписи и таблицы для отображения показателей

процесса. Динамические элементы накладываются на статический фон, меняя свои размеры, форму, цвет или положение в процессе моделирования. При анимации технологических процессов в машиностроении к динамическим элементам относятся заготовки, транспортные средства, исполнительные органы.

Анимационная оболочка Student Proof Animation обрабатывает пять типов входных файлов и шесть выходных. Для анимации простых технологических процессов достаточно использовать два типа входных и три типа выходных файлов (рис. 34).

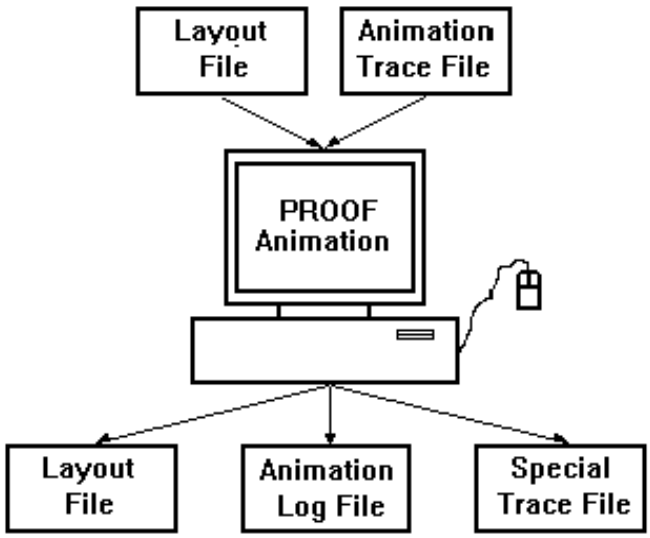


Рис. 34. Входные и выходные файлы,
обрабатываемые Student Proof Animation

Входные файлы:

Layout File – файл разметки. Содержит описание всех статических и динамических объектов: фоновый текст, таблицы, контуры участка цеха, накопители, технологическое оборудование, рабочих и т.д.

Trace File – файл управления анимацией или файл трассировки. Включает команды Student Proof Animation, посредством которых перемещаются динамические объекты.

Выходные файлы:

Layout File – файл разметки создается для сохранения нарисованных графических объектов. Выходной файл разметки подобен входному. После редактирования графических элементов текущее изображение сохраняется как файл с расширением .lay.

Log File – файл регистрации ошибок. Язык Proof Animation отображает на экране предупреждения или сообщения об ошибках. Информация об ошибках автоматически записывается в файле регистрации ошибок (Log File) с расширением .log.

Special Trace File – файл управления анимацией. Управляет движением динамических объектов (рабочих органов, транспортных средств, деталей). Этот файл может генерироваться системой имитационного моделирования GPSS/H.

5.2. Построение статических и динамических объектов в Proof Animation

Для создания статических объектов в Proof Animation используют режим **Draw Mode**. Панель инструментов в Draw Mode имеет следующий вид (рис. 35):

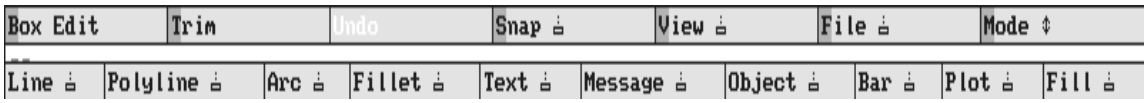


Рис. 35. Панель инструментов Draw Mode

Для создания динамических объектов – классов используют режим Class, в котором имеется точно такая же панель инструментальных средств, как и в режиме Draw. Отличие режима Class от режима Draw в том, что масштаб экрана в режиме Class уменьшен в 10 раз, а точка с координатами (0; 0) находится в центре экрана.

Line *– Добавление линии*

Команду Line используют с соответствующим комплектом инструментальных средств (рис. 36). Построение линии производят указаниями щелчками мыши ее начальной и конечной точки.

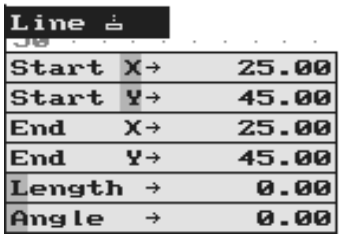


Рис. 36. Комплект инструментальных средств Line

Start X – x – значение начальной отметки Line.

Start Y – y – значение начальной отметки Line.

End X – x – значение конечной отметки Line.

End Y – y – значение конечной отметки Line.

Length – общая длина Line.

Angle – угол, на который Line выводится.

Редактирование линии осуществляют также с помощью команды Line. При редактировании конечные точки линии высвечиваются красными прямоугольниками, а появившееся меню комплекта инструментальных средств Line отразит текущие свойства линии (координаты x и y, длину и т.д.). Чтобы изменить длину или угол наклона линии, необходимо щелкнуть мышью по одной из высвеченных конечных точек и затем переместить ее в новое положение. Для изменения положения линии на экране указатель мыши устанавливают посередине линии и перетаскивают ее в другое место.

Для изменения цвета линии сначала выбирают в цветовой палитре (Color Palette) нужный цвет, а затем щелчком мыши указывают линию.

Polyline *– добавление ломаной линии*

Опция Polyline дает возможность вывести на экран соединенные сегменты Line любой формы. Polyline имеет комплект инструментальных средств, идентичный тому, который был описан в разделе Line. Этот комплект отражает свойства только той линии, которая в настоящее время выводится как часть операции Polyline.

Arc *– добавление дуги*

При использовании Arc комплект инструментальных средств задает следующие параметры дуги (рис. 37):

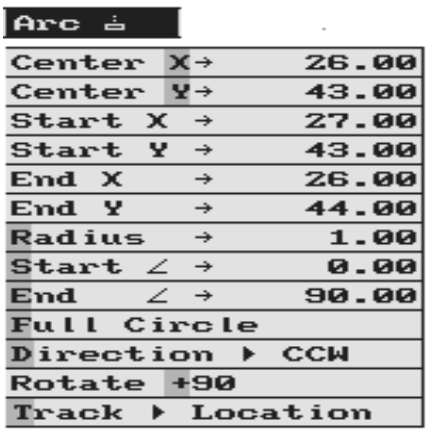


Рис. 37. Комплект инструментальных средств Arc

Center X – значение по x центра Arc.

Center Y – значение по y центра Arc.

Start X – значение по x начальной отметки.

Start Y – значение по y начальной отметки.

End X – значение по x конечной отметки.

End Y – значение по y конечной отметки.

Radius – длина радиуса Arc.

Start – угол от центра до начала отметки.

End – угол от центра до конечной отметки.

Full Circle – опция для преобразования текущей Arc в полный круг.

Direction – направление угла Arc (по часовой стрелке или против часовой стрелки).

Rotate + 90 − инструмент, который вращает Arc на 90˚ по часовой стрелке.

Track – атрибут Arc (Location (Расположение), Radius (Радиус), или Angles (Углы) - могут управляться мышью).

Fill *– заполнение контура цветом*

Fill позволяет заполнить ограниченную область текущим цветом. Ограниченная область может быть задана любой комбинацией Lines и Arcs. Область должна быть замкнута, иначе заполнится все неограниченное пространство. Для изменения текущего цвета области выбирают необходимый цвет в Color Palette и указывают щелчком мыши область заполнения. Для отмены указанных действий используют Undo или Unfill.

Box Edit *– редактирование нескольких элементов*

Box Edit – это мощный инструмент для редактирования группы элементов. Чтобы выбрать элементы, которые нужно отредактировать, обведите эти элементы рамкой блока.

При использовании Box Edit (редактирование блока) появится следующее меню комплекта инструментальных средств (рис. 38):

Cut – удаляет выбранные элементы, после этого они доступны для вставки.

Paste – вставка вырезанных или скопированных элементов.

Name – определит, как обработать именованные элементы при вставке многократных копий.

Delta X – относительное смещение по оси x.

Delta Y – относительное смещение по оси y.

Scale – масштаб.

Rotation – относительный угол вращения.

Vertical Rip – инвертирует элементы «верхняя часть к нижней части».

Horizontal Flip – инвертирует элементы «слева направо».

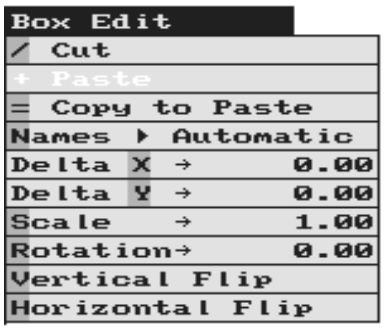


Рис. 38. Комплект инструментальных средств Box Edit

Блок задают щелчками мыши сначала одного, а затем другого противоположного по диагонали угла. Следует обратить внимание на то, что Box Edit можно использовать для перемещения геометрических элементов в Draw Mode и Class Mode, из режима в режим, из файла в файл (изменяя файлы размещения без выхода из Proof Animation).

5.3. Создание классов в Proof Animation

При создании объекта в файле анимации необходимо определить его *Object Class* (класс объекта), который устанавливает форму и другие начальные свойства объекта.

Если в компьютерной анимации применяется множество подобных объектов, то можно задать один или несколько классов объектов. Свойства объекта копируют с «родительского» Object Class, когда объект создан, но любая принадлежность к классу может быть отменена для любого данного объекта.

Определение класса подобно рисунку фонового режима со следующими исключениями:

классы меньше, чем фоновый рисунок;

классы имеют «горячую» и «тыльную» точки;

классы не могут содержать графики и диаграммы.

Class Mode

Панель инструментов в Class Mode имеет следующий вид (рис. 39):

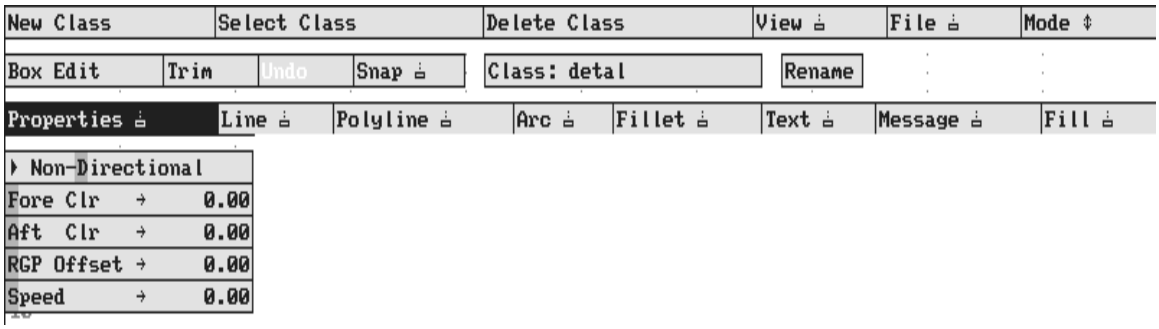


Рис. 39. Панель инструментов в Class Mode

***New Class** (Новый класс)*

New Class используют для определения нового класса. После задания New Class Proof Animation выдает подсказку для ввода имени нового класса. Имена классов чувствительны к регистру и могут быть длиной до 16 символов. Например, имя «FULLAGV» отличается от имени «FullAGV».

Если форма и свойства нового класса определены, то можно продолжать действия, т.к. класс автоматически включается в схему. Однако новые классы не сохраняются на диске, пока не будет задана команда **Save Layout** из меню **File**.

***Select Class** (Выбор класса)*

Опция Select Class позволяет редактировать геометрические элементы или связи существующего класса. При выборе Select Class в центре экрана появится список всех существующих классов для текущего размещения. Выбранный класс становится текущим.

***Delete Class** (Удаление класса)*

Команда Delete Class удаляет с подтверждением действий текущий класс из схемы. Чтобы отменить удаление класса, надо вновь открыть предварительно сохраненную копию вашего файла, не сохраняя при этом последние изменения.

Определение свойств класса

После задания имени класса в верхнем левом углу экрана появляется меню комплекта инструментальных средств **Properties** (свойства). С помощью этого меню могут быть определены следующие четыре свойства класса: *directionality (направление)*, *clearance (диспетчерское разрешение)*, *RGP offset (смещение)* и

speed (скорость). Указанные свойства могут быть определены как до, так и после создания рисунка.

***Hot Point** (Горячая отметка)*

В Class Mode Proof Animation рассматривает, перемещает и вращает объекты, созданные из класса, относительно горячей отметки. Горячая отметка Object Class имеет координаты (0, 0). Для изменения координат горячей отметки используют Box Edit (редактирование блока).

***Directionality and Orientation** (Направление и ориентация)*

Object Class может быть задан как направленный или как ненаправленный. Первая опция в меню **Properties** задает режимы **Non-Directional** (ненаправленный) и **Directional** (направленный). По умолчанию устанавливается **Non-Directional**. Если установлен класс с направленными объектами, то указывают направления перемещения объектов. Если класс с ненаправленными объектами, то объекты перемещают по экрану в одном направлении. Для указания направления можно использовать команду **Box Edit**, которая позволяет вращать геометрию класса.

***Clearance** (Диспетчерское разрешение)*

Для того чтобы объекты, выстроенные в линию, не сталкивались на пути, необходимо определить значение «диспетчерского разрешения». Диспетчерское разрешение наследуется объектом из класса.

Диспетчерское разрешение определяется в линейных единицах (по умолчанию – ноль). Имеются два значения диспетчерского разрешения: «fore» (перед) и «aft» (после). Значения диспетчерского разрешения созданных объектов могут быть изменены командой Object...Clearance (объект набора...Clearance).

***RGP Offset** (RGP смещение)*

RGP смещение позволяет определять факультативную тыльную отметку для текущего класса. RGP действует как вторая отметка присоединения, когда направленные объекты двигаются по пути. RGP смещения должен быть установлен как отрицательное число, поскольку отметка RGP всегда находится позади горячей отметки.

***Speed** (Скорость)*

Все объекты перемещаются с определенной скоростью по пути. Скорость может быть задана для объекта или класса объектов. Скорость класса будет полезна только в тех случаях, когда скорости

несходных объектов на пути зависят от классов объектов, например для транспортных средств, перемещающихся по скоростной дороге.

***Color** (Цвет)*

Цвет – собственность класса. Каждый объект класса может иметь любую комбинацию цветов. Для изменения цвета элемента класса сначала щелчком мыши указывают элемент, а затем цвет. Для замены цвета класса сначала выделяют блок вокруг этого класса командой Box Edit, а затем устанавливают цвет из цветовой палитры.

5.4. Движение в Proof Animation

Большинство видов движения можно представить как перемещение по заданному пути. Задание движения по определенному пути – одна из наиболее важных особенностей Proof Animation. Путь задается по линиям (Lines) и дугам (Arcs) и является невидимым элементом.

***Path Mode** (Режим пути)*

Proof Animation обеспечивает специальный режим определения пути, который можно выбирать из меню **Mode** (режим). Пути показываются в цвете. Другие элементы отображаются черным цветом на сером фоне. В большинстве случаев пути показываются синим цветом. Путь, который необходимо исправить (Repairs), показывается белым цветом. У текущего выбранного пути цвет желтый, а у выбранного сегмента пути цвет красный. Меню **Path Mode** имеет вид, представленный на рис. 40.

Полоса меню Path Mode (режим пути) расположена в верхней части экрана. Она содержит следующие опции: **New Path** (новый путь), **Select Path** (выбрать путь), **Delete Path** (удалить путь), **Repairs** (исправить) и уже знакомые **View** (вид), **File** (файл) и опции **Mode** (режим).

Команда **Delete Path** доступна только после того, как путь был задан или выбран для редактирования. Команда Repairs доступна, если изменения, сделанные в Draw Mode (вывести режим), испортили сегменты одного или более путей.

После выбора команды New Path (новый путь) Proof Animation просит ввести уникальное имя пути. Имя пути может содержать

символы верхнего и нижнего регистров и(или) цифры. Имя должно начинаться с символа и может включать до 16 символов.

После ввода имени пути можно начинать строить схему пути путем добавления сегментов.

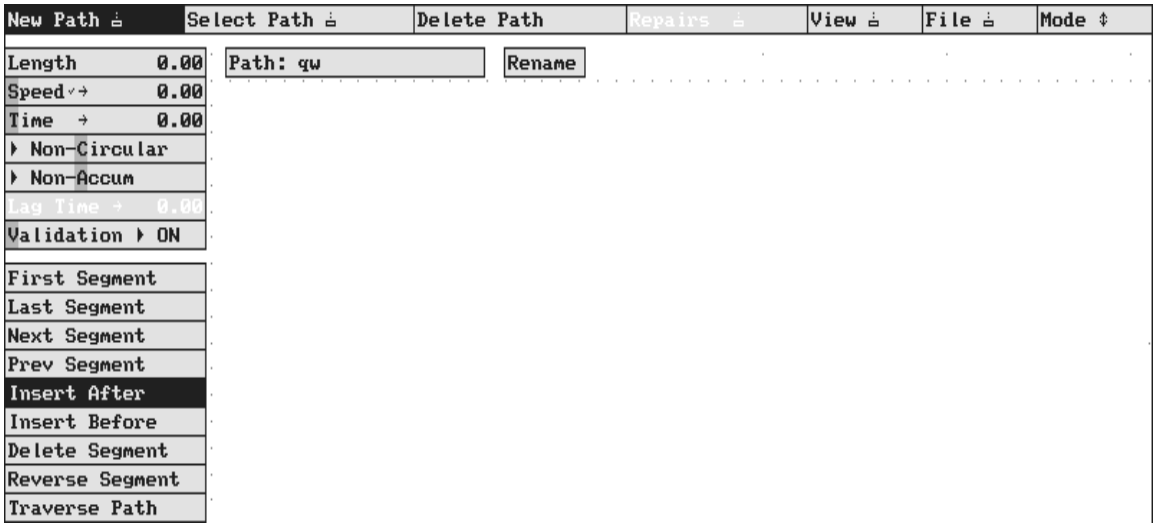


Рис. 40. Панель инструментов в Path Mode

Path Properties *(Свойства пути)*

Верхнее меню комплекта инструментальных средств отображает несколько свойств пути: **Length** (длину), **Speed** (скорость), **Time** (время), **Circularity** (циркуляцию), **Accumulation** (накопление), **Lag Time** (запаздывание) и **Validation** (проверку).

Length *(Длина)*

Proof Animation автоматически определяет длину пути в своих единицах измерения. Длину пути рассматривают как расстояние по пути от начала первого сегмента до конца последнего.

Speed *(Скорость)*

Speed задает скорость двигающихся по пути объектов. После определения скорости на комплекте инструментальных средств появится метка, следующая за Speed. В этом случае время вычисляется и отображается автоматически и основывается на определенной скорости и расчетной длине. Например, если длина пути и скорость равны соответственно 75,0 и 5, то время равно 15,0. Заданная скорость остается такой же для каждого последующего пути, пока скорость или время не будут изменены.

***Time** (Время)*

Командой Time задается время прохождения объекта от начала первого сегмента до конца последнего. Если время определено, то на комплекте инструментальных средств появится маленькая метка, следующая за словом Time. В этом случае скорость вычисляется и отображается автоматически и основывается на определенном значении времени и расчетной длине. Например, если длина равна 75,0, а время равно 150, то скорость будет равна 0,5.

***Accumulation** (Накопление)*

Текущая установка Accumulation переносится на каждый последующий определяемый путь. Значение по умолчанию в Path Mode – «Non – Accum» (ненакопляемый). Накопляемый путь отличается от ненакопляемого обработкой объектов, которые достигли конца пути.

Если путь накопляемый, то объект движется к концу этого пути, чтобы «накопиться» позади других объектов, уже достигших конца этого пути. Когда первый объект удален из накапливаемого пути, другие объекты будут двигаться вперед автоматически. Накопляемый путь не может быть круговым.

***Circularity** (Циркуляция)*

У кругового пути нет конца. Объекты, которые достигают конца пути, переходят непосредственно к началу, даже если конец не соединен с началом. Круговой путь не может быть накапливаемым. Для круговых путей установка значения по умолчанию – «Off» (выключен). Текущая установка переносится к каждому последующему определяемому пути.

***Validation** (Проверка правильности)*

Validation выдает предупреждение Leapfrog, если объект удален из накопляемого пути, или предупреждение Encroachment, если объекты на накопляемом пути сталкиваются друг с другом. Опцию Validation можно переключить на **Validation * ON** (включено) или **Validation * OFF** (выключено).

***Path Segments** (Сегменты пути)*

Пути в Proof Animation могут состоять из одного или более сегментов. Сегмент может относиться ко всей Line (линии) или Arc (дуге) либо к их частям. Сегменты пути упорядочены. В каждом пути имеется первый и последний сегменты. Каждый сегмент имеет направление. Направления сегментов обозначены на экране стрел-

кой в конце каждого сегмента. Эти стрелки не видны в режимах Draw Mode и Run Mode. Конец сегмента не маркирован.

5.4.1. Комплект инструментальных средств создания и редактирования сегментов

В нижнем меню комплекта инструментальных средств есть опции добавления и удаления сегментов. Для добавления сегментов к пути используют опцию **Insert After** (вставка после). Каждый новый сегмент начинается с конца предшествующего сегмента, соединенного с началом стрелки, и указывается щелчком мыши. При задании начала нового пути опция Insert After вызывается автоматически. Для исправления ошибок построения пути из сегментов используют клавиши <Backspace (возврат на один символ) > или <Delete (удалить)>.

Insert Before *(Вставка перед)*, ***Insert After*** *(Вставка после)*

Если сегмент выбран, то в путь можно вставлять один или более сегментов перед ним, нажимая Insert Before (вставка перед), или после него, нажимая Insert After (вставка после). Для нового сегмента пути Insert After выбирается автоматически.

Delete Segment *(Удалить сегмент)*

Delete Segment позволяет удалить выбранный сегмент. После того как сегмент удален, выбирается предыдущий сегмент. Удалять сегменты можно неоднократно, вплоть до достижения начала пути. Если предыдущий сегмент отсутствует, то выбирается следующий сегмент. Можно удалять выбранные сегменты, используя клавиши <Backspace> или <Delete>.

Reverse Segment *(Направить сегмент)*

Reverse Segment позволяет задавать направление выбранного сегмента. Reverse Segment полезен в том случае, если сегмент, особенно первый сегмент пути, направлен неправильно и необходимо повторно выбрать сегмент. Для добавления сегментов после их изменения необходимо сначала нажать Insert After.

Traverse Path *(Двигаться по пути)*

Traverse Path позволяет просмотреть последовательно весь путь по сегментам. Для выхода из Traverse Path используют <Esc>.

5.5. Файл трассировки

Управление динамическими объектами в Proof Animation осуществляется посредством файла трассировки (Trace File). Файл трассировки содержит упорядоченную последовательность команд, в соответствии с которыми Proof Animation выполняет определенные действия в анимационной модели. Такой список команд может быть написан имитационной моделью (GPSS/H-программой). Для написания файла управления вручную можно использовать текстовый редактор, создающий не форматируемые текстовые файлы (например, встроенный редактор NC). Далее рассмотрим основные команды файла трассировки.

Время компьютерной анимации

Команда TIME является основной командой файла трассировки компьютерной анимации в Proof Animation. Синтаксис команды TIME:

TIME <значение времени>

Команду TIME используют для того, чтобы задать часам компьютерной анимации новое значение. Proof Animation продолжает отображать анимацию до тех пор, пока время, определенное в команде TIME, не будет достигнуто. После этого Proof Animation читает и обрабатывает следующую строку в файле следа компьютерной анимации.

Создание объекта

Основным элементом в компьютерной анимации является Object (объект). Объект – это динамический элемент, который может перемещаться на экране. Над объектом выполняют следующие действия: создание, уничтожение, помещение в координатную точку, изменение цвета, изменение формы, перемещение по пути, перемещение от одной отметки до другой, изменение быстродействия, вращение.

При создании объекта вводится его спецификация – класс объектов. Это удобно, поскольку объект может наследовать свойства класса.

Для имен Object Class устанавливают некоторые простые правила. Верхние и строчные символы букв обрабатывают как уникальные символы в именах идентификатора. Каждое имя должно

начинаться с буквенного символа (A-Z или a-z). Имена могут быть длиной не более шестнадцати символов.

Перед тем как объект появится на экране, его сначала нужно создать в файле следа анимации, для этого используют команду Create.

Синтаксис команды Create:

CREATE <имя класса> <номер объекта>

Например: *CREATE Car 20*

Имя класса должно содержать имя класса объектов, который сохранен в файле размещения (*.lay файл). Номер объекта является номером или именем Object (объекта). Номер и имя должны быть уникальными.

Размещение объекта на экране

Proof Animation имеет несколько команд, которые заставляют Object появляться на экране. Основной является команда типа - PLACE ... AT .

Синтаксис PLACE ... AT:

PLACE <номер объекта> AT x y

Например: *PLACE 52 AT 12.5 25*

PLACE ... AT заставляет Object появляться в системе координат Proof Animation, но не заставляет его двигаться.

Удаление объекта

Удаление объектов с экрана осуществляется командой DESTROY. При этом объект удаляется только с экрана, а на класс объектов данная команда не воздействует.

Синтаксис для DESTROY:

DESTROY <имя объекта>

Установка и изменение цвета объекта

Если объект уже видим на экране, можно изменить его цвет при помощи команды SET COLOR в файле следа.

Синтаксис команды SET COLOR:

SET <имя объекта> COLOR

В Proof Animation используются цвета двух типов:

- Цвета фона:

BACKDROP (Black)

LAYOUT (Lay)

- Цвета активного слоя:

BLUE или F1

RED – F2
WHITE – F3
YELLOW – F4
PINK – F5
TAN – F6
GREEN – F7

Для определения цвета используется соответствующий номер с буквой F или английское название цвета. Например,

SET 3 COLOR F2
SET BOX COLOR RED

Объекты фоновых цветов двигаются без интерференции через другие объекты. Если два объекта цвета активного слоя соприкоснутся друг с другом, то область перекрытия будет показана в третьем отличном цвете. Если произойдет наложение двух одинаковых объектов одного активного цвета, то появится цвет самого нижнего слоя BACKDROP (объекты станут невидимы).

Перемещение объектов между двумя точками

Движение по прямой между двумя точками применяется для перемещения объектов в точку, координаты которой известны или могут быть вычислены в ходе выполнения имитационной модели. Для такого перемещения используется команда MOVE.

Синтаксис команды MOVE:

MOVE <имя объекта> (скорость движения) (x) (y) [RELATIVE]

При выполнении этой команды объекты двигаются по прямой линии между двумя точками. Объект сразу начинает движение из текущих координат к точке с координатами (x, y) с заданной скоростью. Достигнув заданных координат, он останется в этой точке на время, соответствующее модулю времени, определенному в команде TIME. Если используется ключевое слово RELATIVE, то объект начнет перемещаться из текущих координат на заданное количество модулей в направлении x и y. Например, по команде: *MOVE 21 10 4 6 RELATIVE* объект 21 начнет перемещаться из текущих координат на 4 единицы в направлении x и на 6 единиц в направлении y со скоростью 10.

Рассмотрим еще один пример. Допустим, что объект робокар «rob» создан. Тогда в результате выполнения подпрограммы:

TIME 50

MOVE rob 10 40 50
TIME 60

по достижении текущего времени TIME 50 объект «rob» начнет перемещаться со скоростью 10 в точку с координатами x = 40, y = 50. Его перемещение будет продолжаться 60 – 50 = 10 единиц времени.

Перемещение объектов по пути

Если Object помещен на Path и быстродействие Path-и или Object-а больше чем ноль, Object автоматически будет двигаться по Path-и, пока не достигнет конца, если некоторая другая команда файла следа компьютерной анимации (например, DESTROY или END) не прервет движение.

Команда для использования Paths – PLACE ... ON. Синтаксис для PLACE ... ON:

PLACE <номер объекта> ON <имя пути>

Например: *PLACE Box ON Conveyor*

Вращение Объектов

Если объект помещен на экран, его можно вращать. Причем это вращение не зависит от действий над объектами, т.е. во время вращения можно изменять цвет объекта, перемещать его командой MOVE или помещать на путь. Для вращения объектов используется команда ROTATE.

Синтаксис команды:

ROTATE <имя объекта> [TO] <угол> TIME <время> <шаг>

ROTATE <имя объекта> TO <угол> SPEED <скорость> <шаг>

Угол поворота и шаг измеряются в градусах, продолжительность поворота в единицах времени Proof Animation, скорость поворота в градусах на единицу времени. Положительный угол – против часовой стрелки, а отрицательный – по часовой. Если задать угол, но не задавать продолжительность и скорость поворота, то вращение произойдет мгновенно. Если определить скорость поворота, но не определять угол, то объект будет вращаться с этой скоростью до выполнения другой команды ROTATE или до отмены этой команды. Использование ROTATE со скоростью 0 остановит вращение. Плавность вращательного движения управляется величиной шага. По умолчанию Proof Animation вращает объект с приращением 30˚.

Вращение может быть абсолютным и относительным. При абсолютном вращении в команде ROTATE перед углом ставится опция TO. Если опустить опцию TO в команде ROTATE, то вращение будет относительным.

Пример абсолютного вращения:

ROTATE 1 TO 90

... мгновенно вращает объект 1 на угол 90˚ против часовой стрелки (рис. 41).

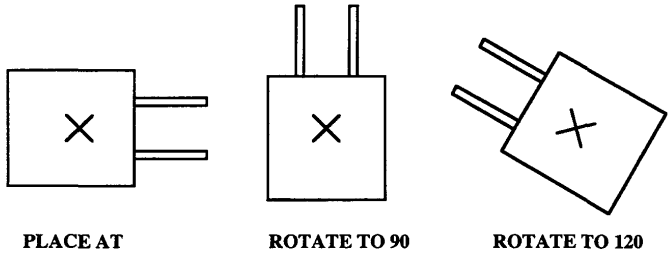


Рис. 41. Пример абсолютного вращения

ROTATE 1 120

... мгновенно вращает объект 1 на 120˚ против часовой стрелки относительно положения, в которое его поместили командой PLACE ... ON.

Пример относительного вращения:

ROTATE 1 90

... мгновенно вращает объект 1 на угол 90˚ против часовой стрелки (рис. 42).

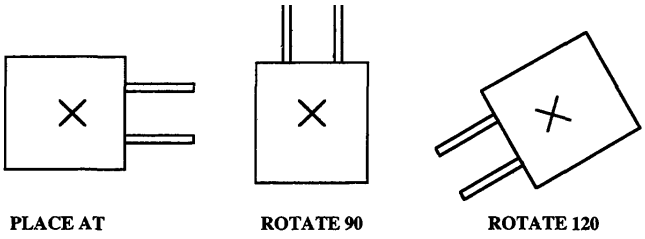


Рис. 42. Пример относительного вращения

ROTATE 1 120

... мгновенно вращает объект 1 на 120˚ против часовой стрелки относительно текущей позиции.

Другие примеры команды ROTATE:

ROTATE 1 SPEED -90

... непрерывно вращает объект 1 по часовой стрелке относительно горячей точки на 360˚ (четыре единицы времени Proof Ani-

mation). Другая команда ROTATE с определенным углом остановит непрерывное вращение.

ROTATE 1 90 TIME 10

ROTATE 1 90 SPEED 9

... эквивалентные команды вращают объект 1 в три шага *(т.к. по умолчанию 30˚)* за 10 единиц времени Proof Animation. Вращение заканчивается, когда объект повернется на 90˚ против часовой стрелки относительно текущей ориентации.

ROTATE 1 90 TIME 10 STEP 5

... то же, что и в предыдущем примере, но объект будет делать 18 шагов вместо трех, для поворота на 90˚ (здесь отменяется заданный по умолчанию размер шага – 30˚).

Изменение содержания сообщения

Прежде чем вывести динамическое сообщение, необходимо его создать и определить его прототип в цвете Backdrop режиме Draw Mode по команде Message. Изменить сообщение (его цвет и содержание) можно по команде WRITE.

Синтаксис команды WRITE:

WRITE <имя сообщения> <текст сообщения>

Например, по команде:

WRITE line1 Control of robocar

в линию line1 (должна быть определена в разметке) помещается надпись «Control of robocar» (Управление робокаром).

Завершение анимации

Файл следа обязательно должен заканчиваться командой END.

Синтаксис команды END:

END

5.6. Пример анимационного моделирования

Процедуру создания анимационной модели покажем на примере изготовления деталей разными типами станков.

Заготовка последовательно обрабатывается двумя типами станков (типом А и В). Группа станков А включает два станка. В начале процесса заготовка в течение 15 единиц времени транспортируется до станков типа А, откуда поступает на обработку на первый станок группы. Здесь происходит ее обработка в течение 10 единиц времени. После этого полуфабрикат в течение 5 единиц

времени поступает на обработку на станок типа B. Окончательная обработка детали занимает 20 единиц времени. Готовая деталь транспортируется в течение 5 единиц в накопитель. После этого к станкам группы A поступает вторая заготовка и процесс повторяется за исключением того, что в группе A заготовка обрабатывается вторым станком. Необходимо создать анимационную модель технологического процесса.

Метод построения модели

Войдем в Student Proof Animation путем загрузки файла spa.exe. Выберем в верхнем меню пункт Mode, а в нем режим Draw Mode. С помощью набора команд Student Proof Animation нарисуем статические элементы технологии: станки, траектории движения деталей (в невидимом цвете Backdrop) и необходимый текст (рис. 43).

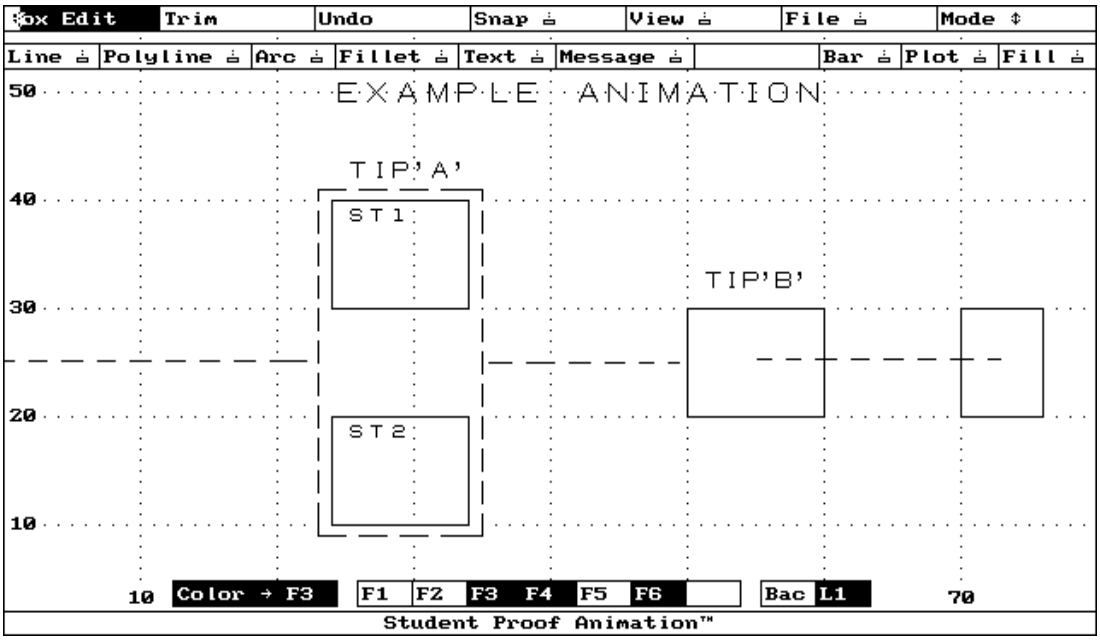


Рис. 43. Статические элементы технологии
(пунктиром показаны траектории движения динамических элементов, в режиме Draw они невидимы)

В режиме Class Mode выберем опцию «New class» и нарисуем динамический объект (класс) – деталь (под именем det) (рис. 44).

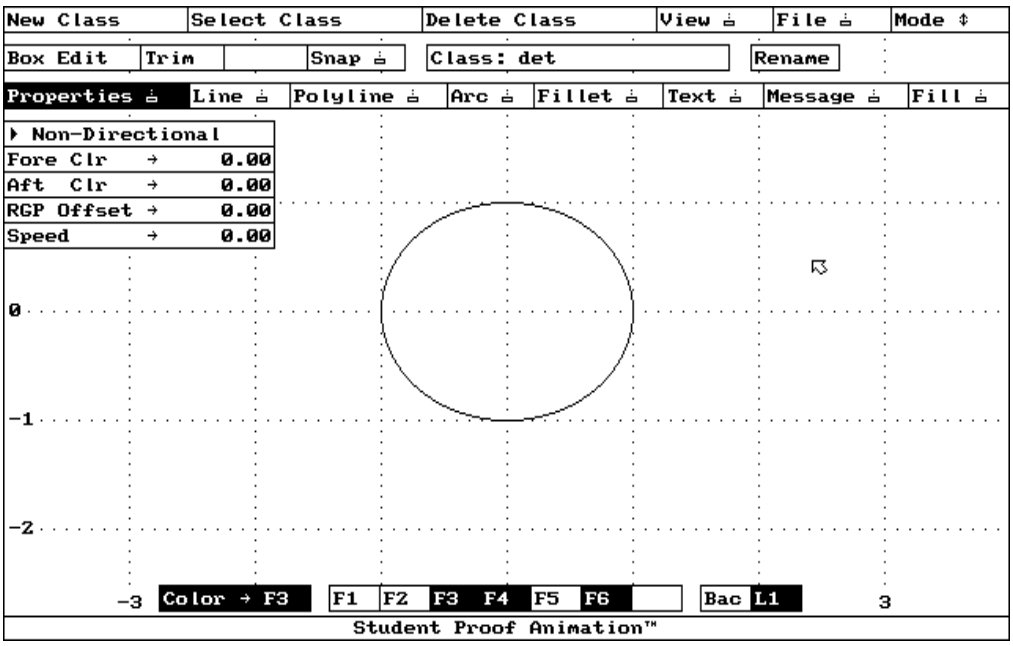


Рис. 44. Динамический объект технологии

В режиме Path Mode выберем опцию «New Path», создадим пути на основе уже нарисованных (в режиме Draw) траекторий движения деталей и назовем их p1, p2 и p3. Выберем в меню опцию «Time» и впишем время движения по пути, равное 15 для p1 и 5 для p2 и p3 (рис. 45).

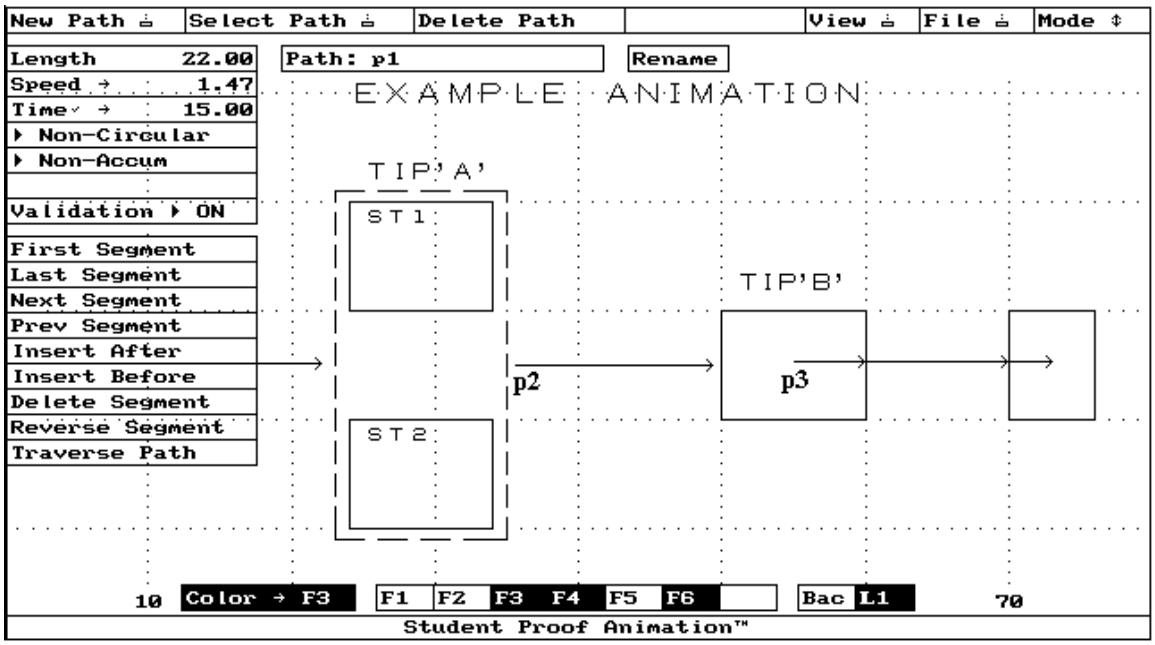


Рис. 45. Пути движения динамических объектов

Сохраним файл разметки как primera.lay и выйдем из Proof Animation.

При помощи текстового редактора (Far, NC, VC, Блокнот, WordPad и т.п.), формирующего ASCII файлы, используя команды Student Proof Animation для управления динамическими объектами, создадим файл трассировки с тем же названием, что и файл разметки (*.lay файл), но с расширением *.atf – primera.atf. Листинг файла трассировки приведен ниже:

```
TIME 0
CREATE det 1
PLACE 1 ON p1
TIME 15
PLACE 1 29 15
TIME 25
SET 1 COLOR YELLOW
PLACE 1 ON p2
TIME 30
PLACE 1 55 25
TIME 50
SET 1 COLOR RED
PLACE 1 ON p3
TIME 55
CREATE det 2
PLACE 2 ON p1
TIME 70
PLACE 2 29 35
TIME 80
SET 2 COLOR YELLOW
PLACE 2 ON p2
TIME 85
PLACE 2 55 25
TIME 105
SET 2 COLOR RED
PLACE 2 ON p3
TIME 110
END
```

Сохраним этот файл в той же директории, что и файл разметки.

После сохранения файла управления (как primera.atf) запустим анимацию, выбрав файл spa.exe и нажав клавишу <Enter>. Выведем на экран анимационную модель технологии путем входа с помощью мыши в пункт «File» верхнего горизонтального меню, подпункт «Open Layout&Trace». В появившемся вертикальном меню выбе-

рем пункт primera путем нажатия на нем клавиши мыши. В режиме «RUN» командой «Go» запустим программу на выполнение.

На рис. 46 представлен фрагмент анимации технологии изготовления деталей.

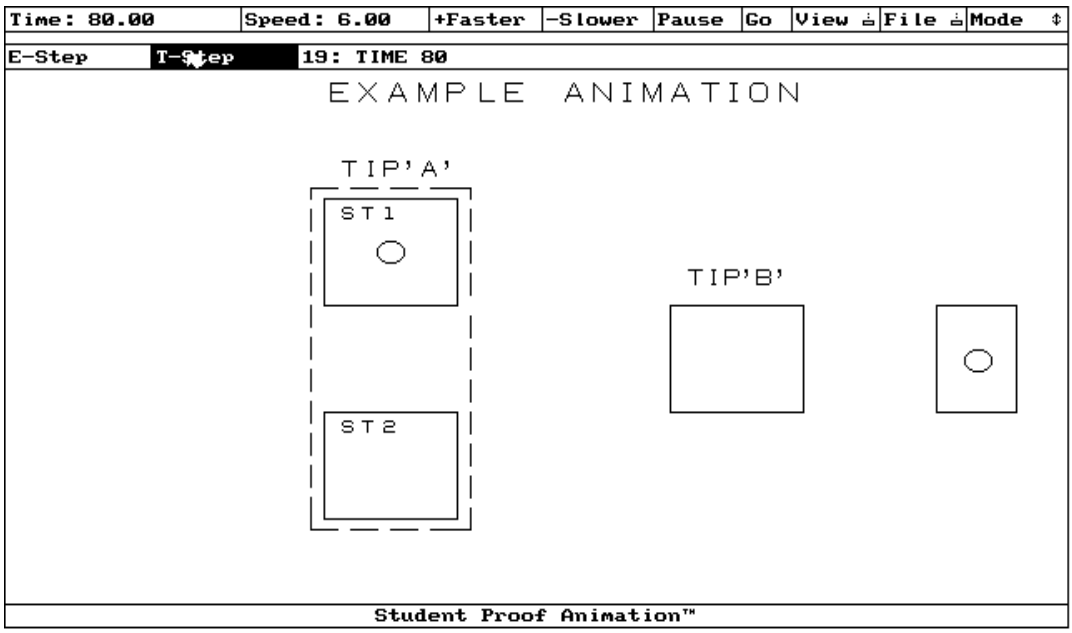


Рис. 46. Фрагмент анимации технологии изготовления деталей

6. СВЯЗЬ АНИМАЦИИ С ИМИТАЦИОННОЙ МОДЕЛЬЮ

6.1. Генерирование файла трассировки (.atf) имитационной моделью

Управляющим оператором и блоком, создающим линии файла трассировки, являются оператор PUTPIC и блок BPUTPIC.

Формат:

PUTPIC opt,..., (list)

BPUTPIC opt,..., (list)

где opt – опция, а list – список чисел, числовых выражений, переменных, стандартных числовых атрибутов, которые GPSS/H записывает во внешний файл (файл.atf).

Опции, связанные с оператором PUTPIC и блоком BPUTPIC:

FILE=log

LINES=unt

где log – логическое имя внешнего файла, в который будут записываться данные (по умолчанию данные будут выводиться на экран), а unt – число строк, отображаемых после блока BPUTPIC.

Для создания GPSS/H-модели файла управления анимацией, сначала его необходимо связать с логическим именем, которое будет использоваться в GPSS/H-программе. Для такой связи используется специальный оператор FILEDEF.

Формат оператора FILEDEF:

LOGFILEDEF 'NAME'

где NAME – имя .atf-файла, а LOG – логическое имя файла.

Например,

```
...
ATF   FILEDEF    «TIME.ATF»
...
BPUTPIC    FILE=ATF,LINES=3,AC1
TIME *.**
CREATE rob rob
PLACE rob at 12 18
```

В приведенном примере файл «TIME.ATF» будет автоматически создан в текущей директории, связан с логическим именем ATF и в него будут записаны три строки, расположенные ниже блока BPUTPIC. При этом вместо звездочек (*.**) запишется значение текущего времени моделирования (значение стандартного числового атрибута AC1).

Для записи в файл управления команды END используется управляющий оператор PUTPIC:

```
PUTPIC    FILE=ATF
END
```

Такая запись обязательна, она используется для завершения анимации и обычно записывается после оператора START.

6.2. Переменные в GPSS/H-моделях

При связи GPSS/H-модели с внешними файлами используют переменные. Переменные позволяют читать из внешнего файла входные параметры и записывать во внешний файл (.atf) результаты моделирования.

Для определения переменных в GPSS/H-моделях используются утверждения INTEGER (целая) и REAL (действительная).

Формат INTEGER и REAL:

INTEGER &v, . . .

REAL &v, . . .

где v – имя переменной.

При объявлении переменных их начальные значения равны 0. Для присвоения численных значений, отличных от нуля, используются оператор LET и блок BLET.

Формат блока BLET и оператора LET:

LET &v=var

BLET &v=var

где v – имя переменной, а var – числовое значение или математическое выражение.

Предположим, что переменные &TIME, &V и &S определены в модели как действительные

REAL &TIME,&V,&S

и обозначают соответственно время движения транспортного средства от склада до рабочего места, скорость движения и расстояние доставки соответственно. Тогда для присвоения переменной &TIME числового значения можно использовать строку

LET &TIME=&V/&S

Примечание

Для получения нецелого результата, например в выражении 3/2, необходимо значения числителя и знаменателя записывать через десятичную точку, т.е. 3.0/2.0. В противном случае из результата выражения будет отброшена дробная часть.

6.3. Чтение данных из внешнего файла

Оператор GETLIST – читает данные из внешнего файла или с клавиатуры (данные во внешнем файле должны быть в свободном формате, с входными значениями, отделяемыми одним или более пробелами или написанными на разных строках).

Формат оператора GETLIST:

GETLIST FILE=(логическое имя файла),(list),…

где FILE – опция, определяющая логическое имя файла, из которого должны читаться входные данные, а list – список переменных, которым будут присвоены числовые значения.

Например, в GPSS/H-модели значение переменной &I, обозначающей количество станков в цехе, должно читаться из внешнего файла «IN». Этого можно добиться, записав в модели строку:

```
GETLIST   FILE=IN,(&I)
```

6.4. Пример связи анимации с имитационной моделью

Для демонстрации связи анимации с имитационной моделью воспользуемся примером имитационного моделирования, приведенным в п. 4.5, и примером анимации в п. 5.6 данного учебного пособия.

Свяжем имитационную и анимационную модели технологического процесса изготовления деталей, причем так, чтобы входные данные вводились из внешнего файла, а результаты моделирования выводились на монитор компьютера.

Метод построения модели

Создадим файл входных параметров под именем data, используя текстовый редактор, например, оболочки NC5.0. Текст этого файла приведен ниже.

```
**********************************************************
28      *       Интервалы поступления заготовок, мин
32      *       Время движения заготовок до станков типа A, мин
10      *       Отклонение от среднего времени, мин
63      *       Средняя прод-ть обработки заготовки на ст. типа A, мин
9       *       Отклонение от средней прод-ти, мин
12      *       Среднее время движения до станка типа B, мин
5       *       Отклонение от среднего времени, мин
55      *       Средняя прод-ть обработки детали на станке типа B, мин
5       *       Отклонение от средней прод-ти, мин
50      *       Кол-во деталей, которые необходимо изготовить, шт.
**********************************************************
```

Изменим GPSS/H-модель в соответствии с заданными требованиями:

- для ввода входных параметров модели из внешнего файла используем GPSS/H-переменные (с утверждениями INTEGER и REAL) и оператор GETLIST;

- для перенаправления заготовок к свободному станку группы А добавим в имитационную модель «внутрь» многоканального устройства STA1 два прибора STAN1 и STAN2;

- для формирования файла управления анимацией воспользуемся блоком BPUTPIC совместно с управляющим оператором FILEDEF;

- для вывода результатов моделирования на монитор компьютера используем стандартные числовые атрибуты;

- в места, где требуется анимация операций, введем команды файла управления Student Proof Animation.

После ввода изменений сохраним файл как primer.gps. Измененная программа модели компьютерной имитации и анимации представлена ниже:

```
        SIMULATE
ATF     FILEDEF         'PRIMER.ATF'
        REAL            &POST,&TDSA,&OTDSA,&TOSA,&OTOSA
        REAL            &TDSB,&OTDSB,&TOSB,&OTOSB
        INTEGER         &NDET
        GETLIST         FILE=DATA,(&POST)
        GETLIST         FILE=DATA,(&TDSA)
        GETLIST         FILE=DATA,(&OTDSA)
        GETLIST         FILE=DATA,(&TOSA)
        GETLIST         FILE=DATA,(&OTOSA)
        GETLIST         FILE=DATA,(&TDSB)
        GETLIST         FILE=DATA,(&OTDSB)
        GETLIST         FILE=DATA,(&TOSB)
        GETLIST         FILE=DATA,(&OTOSB)
        GETLIST         FILE=DATA,(&NDET)
        STORAGE         S(STA1),2
        GENERATE        RVEXPO(1,&POST)
        BPUTPIC         FILE=ATF,LINES=3,AC1,XID1,XID1
TIME *.**
CREATE det *
PLACE * ON p1
        ADVANCE         RVNORM(3,&TDSA,&OTDSA)
        QUEUE           LINE1
        ENTER           STA1
        DEPART          LINE1
        TRANSFER        BOTH,,DRUG
        SEIZE           STAN1
        BPUTPIC         FILE=ATF,LINES=3,AC1,XID1,QA(LINE1)
TIME *.**
```

```
PLACE * 29 35
WRITE QUEUE1 Average queue=*.*
        ADVANCE         &TOSA,&OTOSA
        RELEASE         STAN1
        LEAVE           STA1
        BPUTPIC
        FILE=ATF,LINES=5,AC1,XID1,XID1,FR(STAN1)/10,SR(STA1)/10
TIME *.**
SET * COLOR YELLOW
PLACE * ON p2
WRITE UTIL1 Util=*%
WRITE UTIL Util=*%
        ADVANCE         RVNORM(5,&TDSB,&OTDSB)
        TRANSFER        ,OSP
DRUG    SEIZE           STAN2
        BPUTPIC         FILE=ATF,LINES=2,AC1,XID1
TIME *.**
PLACE * 29 15
        ADVANCE         &TOSA,&OTOSA
        RELEASE         STAN2
        LEAVE           STA1
        BPUTPIC
        FILE=ATF,LINES=5,AC1,XID1,XID1,FR(STAN2)/10,SR(STA1)/10
TIME *.**
SET * COLOR YELLOW
PLACE * ON p2
WRITE UTIL2 Util=*%
WRITE UTIL Util=*%
        ADVANCE         RVNORM(5,&TDSB,&OTDSB)
OSP     QUEUE           LINE2
        SEIZE           STA2
        DEPART          LINE2
        BPUTPIC         FILE=ATF,LINES=3,AC1,XID1,QA(LINE2)
TIME *.**
PLACE * 55 25
WRITE QUEUE2 Average queue=*.*
        ADVANCE         &TOSB,&OTOSB
        RELEASE         STA2
BOX     BPUTPIC
        FILE=ATF,LINES=5,AC1,XID1,XID1,FR(STAN2)/10,N(BOX)+1
TIME *.**
SET * COLOR RED
PLACE * ON p3
WRITE UTIL3 Util=*%
```

```
WRITE N *
        TERMINATE       1
        START           &NDET
        PUTPIC          FILE=ATF
END
        END
```

Запустим интерпретатор GPSS/H. В строке запроса имени файла указываем имя *.gps-файла нашей модели (primer.gps) и нажмем клавишу <Enter>. В результате сформируется файл управления анимацией (файл primer.atf) с результатами моделирования.

Примечание

В ходе анимации заявки, двигающиеся по путям и скапливающиеся перед графическими аналогами станков, будут накладываться друг на друга. Для того чтобы этого не происходило, необходимо войти в режим Path, выделить путь p1 и изменить опцию Non-Accum (ненакопляемый) на Accumulating (накопляемый). Затем тоже проделать с путем p2. После этого необходимо войти в режим Class, выбрать класс под именем det и изменить значения Fore Clr и Aft Clr (см. п. 5.4 Движение в Proof Animation).

На рис. 47 представлен фрагмент компьютерной анимации с результатами имитационного моделирования.

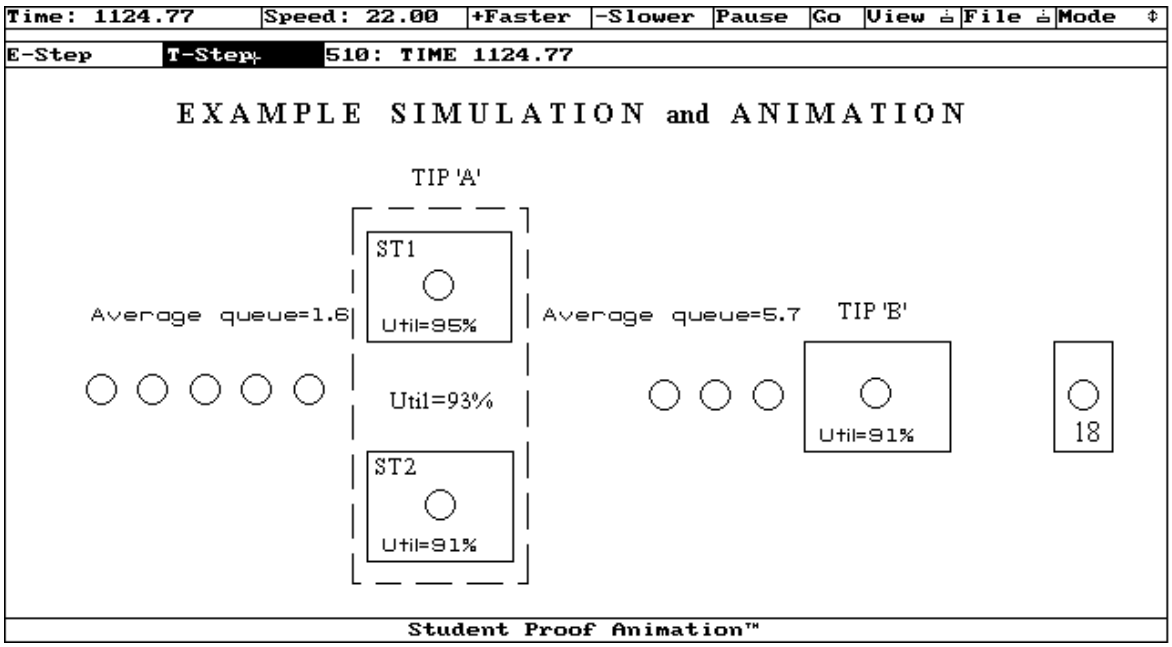


Рис. 47. Фрагмент анимации с результатами имитационного моделирования

7. ЭТАПЫ СОЗДАНИЯ МОДЕЛИ КОМПЬЮТЕРНОЙ ИМИТАЦИИ И АНИМАЦИИ

Создание модели компьютерной имитации и анимации сводится к четырем этапам.

Этап 1. Представление заданного технологического процесса в виде системы массового обслуживания (СМО).

На этом этапе реализуется переход от словесного описания технологического процесса к его математической модели. Здесь требуется описать заданный объект моделирования в абстрактных терминах и понятиях с использованием теории систем массового обслуживания.

Для этого необходимо:

- определить потоки событий (входящие потоки заявок и потоки обслуживаний для каждой очереди и прибора обслуживания);
- определить структуру системы массового обслуживания (число фаз, число каналов обслуживания, число очередей для каждой из фаз обслуживания заявок и связи источников заявок, приборов и очередей);
- определить алгоритмы функционирования системы массового обслуживания (дисциплины ожидания заявок в очередях и выбора на обслуживание каналов, правила ухода заявок из очередей и приборов).

Этап 2. Разработка в соответствии с СМО имитационной модели на специализированном языке GPSS/H.

Здесь математическая модель, сформированная на первом этапе, воплощается в конкретную машинную модель, ориентированную на использование специализированного языка компьютерной имитации GPSS/H. Вначале требуется построить модель по блочному принципу, т.е. в виде совокупности стандартных блоков языка GPSS/H. Для этого необходимо:

- определить, какие объекты технологии будут отображаться транзактами в модели;
- определить количество сегментов, из которых будет состоять модель;
- подобрать блоки (цепь блоков), которые будут отображать события в заданной технологии.

После построения блок-схемы необходимо перейти к программированию модели. Переход от блок-схемы к программе является формальным шагом, так как заключается в записи пространственной структуры в линейном виде с добавлением необходимых управляющих операторов, что не требует специальных навыков.

На данном этапе также рекомендуется создать файл-меню, куда будут вводиться исходные данные. Файл-меню создается специально для данного процесса в виде неформатируемого текстового файла типа ASCII.

Этап 3. Отображение динамики технологического процесса при помощи языка компьютерной анимации Proof Animation.

На этом этапе требуется отобразить динамику заданного технологического процесса на компьютере при помощи специализированного языка компьютерной анимации Proof Animation. Для этого при помощи опций режима Draw Mode необходимо нарисовать статические элементы анимации (контуры цеха, станки, траектории движения транспортных средств и т.п.), а в режиме Class Mode динамические объекты (детали, транспортные средства, рабочих и т.п.).

Для связи анимации с имитационной моделью необходимо добавить в те места GPSS/H-модели, где требуется динамическое отображение операций, специальные команды управления Proof Animation.

На этом этапе также требуется вывести результаты моделирования на анимацию технологического процесса.

Этап 4. Оценка характеристик технологического процесса на разработанной модели компьютерной имитации и анимации.

На этом этапе компьютер используется для проведения имитационных экспериментов на составленной программе. Результаты этих экспериментов надо использовать для анализа и формулирования выводов о характеристиках заданного технологического процесса. Необходимо решить вопрос о форме представления результатов моделирования (графики, диаграммы, гистограммы, схемы и т.п.). В каждом конкретном случае целесообразно выбрать наиболее подходящую форму представления результатов моделирования. В большинстве случаев результаты удобнее сводить в таблицы, хотя графики позволяют более наглядно иллюстрировать полученные результаты.

При проведении имитационных экспериментов сначала надо вызвать файл-меню и ввести в него исходные данные. Затем запустить составленную программу GPSS/H. После прогона модели сформируется файл управления анимацией .atf-файл. При запуске анимации результаты моделирования должны быть выведены на экран в виде движения оборудования, рабочих, надписей и количественных показателей процесса.

8. ПРИМЕР СОЗДАНИЯ МОДЕЛИ КОМПЬЮТЕРНОЙ ИМИТАЦИИ И АНИМАЦИИ

Разработка модели компьютерной имитации и анимации показана на примере, в котором рассматривается процесс обслуживания нескольких рабочих мест робокаром.

Описание объекта моделирования

Робокар обслуживает 5 рабочих мест, расположенных на разном расстоянии от склада (табл. 14). С рабочих мест на склад поступают заявки. Поток заявок является случайным, с равномерным законом распределения. Интервал между поступлениями заявок 120±20 минут. По мере поступления заявок с рабочих мест устанавливается очередность обслуживания. Робокар по заданной программе управления оснащается требуемым грузом, перемещается к рабочему месту, разгружается и возвращается на склад.

Таблица 14

Расстояния от склада до рабочих мест

Рабочее место	Расстояние от склада до рабочего места, м
1	500
2	300
3	100
4	200
5	160

Продолжительность загрузки и разгрузки робокара колеблется в интервале от 6 до 10 мин, распределенном по нормальному закону. Скорость робокара постоянна и равна 30 м/мин.

Задание

Используя методы компьютерной имитации и анимации, определите:

количество обслуженных заявок за 8-часовой рабочий день;

среднюю продолжительность обслуживания заявки;

загрузку робокара;

средний размер очереди на обслуживание из всех рабочих мест;

среднее время ожидания заявки в очереди.

Исследуйте изменение загрузки робокара при изменении числа рабочих мест от 1 до 5.

Оцените изменение среднего размера очереди заявок на обслуживание от числа рабочих мест при скорости движения робокара 30 и 52 м/мин.

Решение

Этап 1. Представление заданного технологического процесса в виде системы массового обслуживания (СМО).

Процесс отображается системой массового обслуживания, в которой заявки, поступающие из *n* рабочих мест, последовательно проходят через обслуживающий прибор – робокар. Поток требований, формируемый рабочими местами, и время обслуживания каждого требования являются случайными величинами. Представим требование на определенный груз как заявку, а робокар – как прибор, выполняющий заявку за определенное время. Тогда технология обслуживания *n*-го количества рабочих мест робокаром может быть описана как однофазная, многоканальная СМО без отказов с прибором, выполняющим заявки на загрузку, перемещение и разгрузку определенного груза (рис. 48).

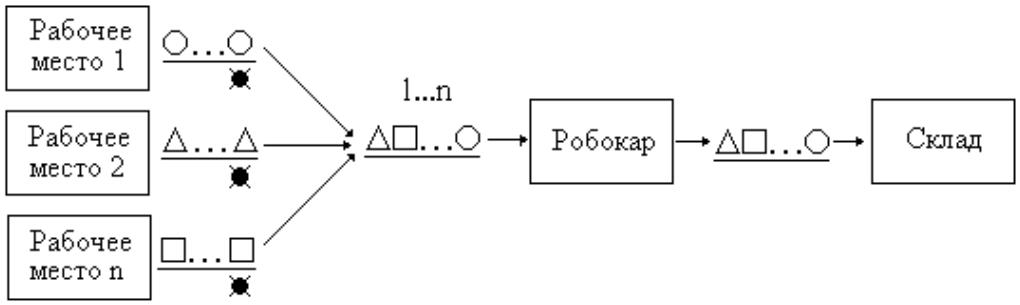


Рис. 48. Процесс обслуживания рабочих мест робокаром в виде СМО

Этап 2. Разработка в соответствии с СМО имитационной модели на специализированном языке GPSS/H.

Переходя от системы массового обслуживания к специализированному языку компьютерной анимации GPSS/H, определим, что заявки в СМО соответствуют транзактам в имитационной модели. Для их ввода в модель используем блоки GENERATE. Так как по условию задания необходимо моделировать 5 рабочих мест, то модель построим из 5 сегментов, которые отображают поступление заявок с каждого рабочего места. Для перенаправления заявок на обслуживание робокаром используем блок TRANSFER в режиме безусловной передачи. Прибором в модели является робокар с именем ROB. Его описываем цепью блоков SEIZE-RELEASE. Для моделирования продолжительности операций загрузки, разгрузки и перемещения робокара используем блоки ADVANCE с распределением временных интервалов согласно заданию. Продолжительность рейса робокара зависит от продолжительности загрузки, разгрузки и движения до рабочего места и обратно. Так как расстояния до рабочих мест разные, то заявки должны обслуживаться за разное время, т.е. в модели необходимо ввести различие между заявками с разных рабочих мест. Для этих целей воспользуемся блоками ASSIGN, при помощи которых в параметры транзактов действительного типа (REAL) будут записываться тип транзакта и продолжительность движения в зависимости от расстояния до рабочего места. Значения, записанные в эти параметры, будут использоваться затем в модели для имитации рейсов робокара. Для отображения выполненной заявки используем блок TERMINATE, который удаляет транзакты из модели. Для имитации 8-часового рабочего дня в модель введем дополнительный сегмент, состоящий из двух блоков GENERATE и TERMINATE.

Разработанная в соответствии с СМО блок-схема имитационной модели процесса обслуживания рабочих мест робокаром приведена на рис. 49. Она содержит 31 блок в 6 сегментах.

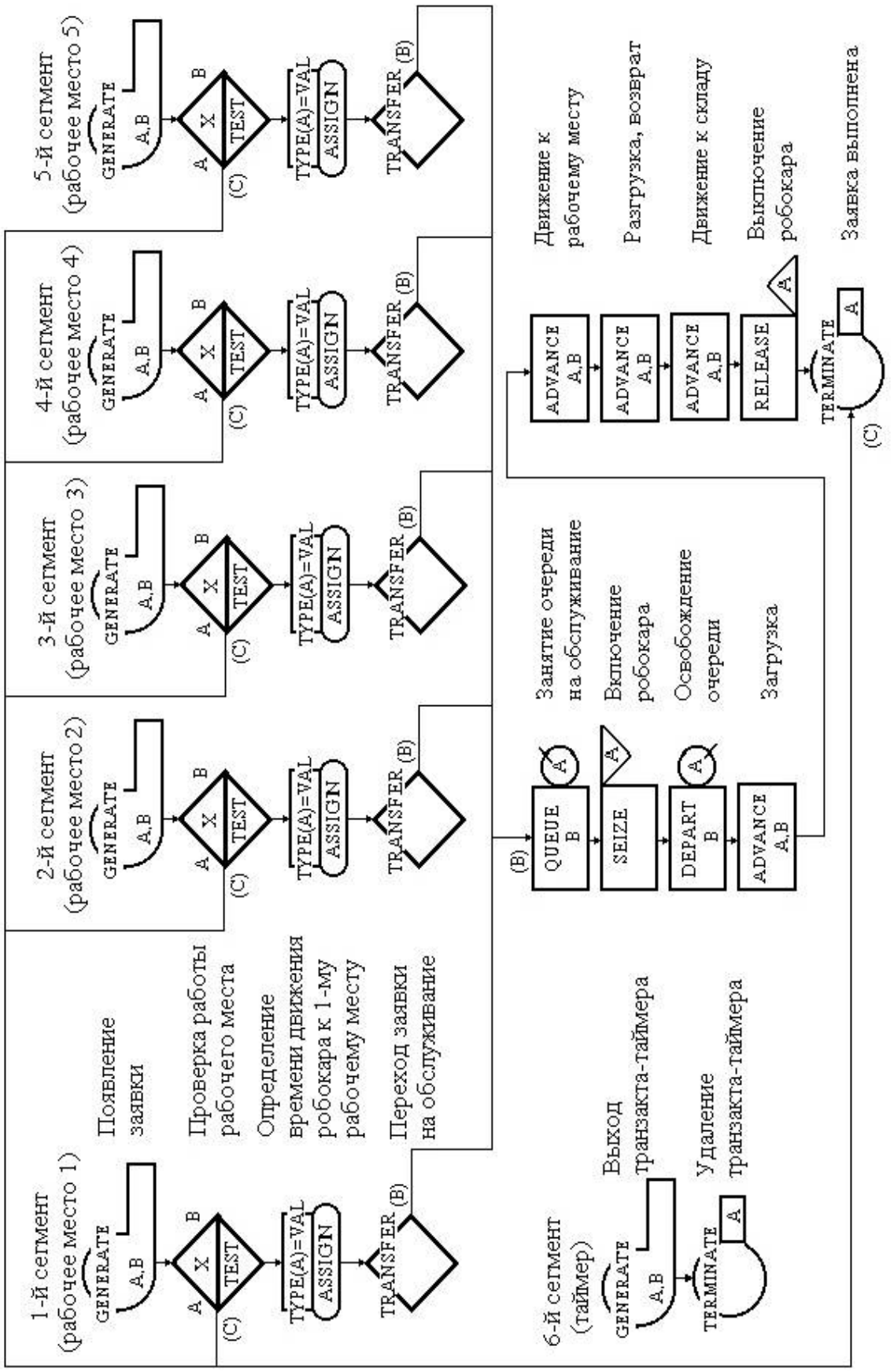


Рис. 49. Блок-схема GPSS/H-модели процесса обслуживания рабочих мест робокаром

GPSS/H- программирование

Для ввода исходных данных в модель, при помощи текстового редактора создан файл-меню. Часть этого файла представлена ниже (полная распечатка файла-меню приведена в прил. 1).

```
**************************************************************
* Исходные данные для модели процесса обслуживания рабочих мест робо-
каром
**************************************************************
*    Значение          *          Тип данных
**************************************************************
1                  *Рабочее место 1 (включено 1; выключено 0)
1                  *Рабочее место 2 (включено 1; выключено 0)
.
.
8                  *Средняя продолжительность разгрузки робокара, мин
2                  *Отклонение от средней продолжительности, мин
480                *Продолжительность моделирования, мин
**************************************************************
```

А – Создание .atf-файла и определение переменных

В соответствии с разработанной блок-схемой написана программа на языке GPSS/H, часть которой представлена ниже (полная распечатка программы приведена в прил. 2). В программе можно выделить несколько частей:

```
ATF  FILEDEF   'ROBO.ATF'                          Создать файл
                                                    robo.atf
INTEGER  &M1,&M2,&M3,&M4,&M5,&SHIFT                Определение целых
                                                    переменных
REAL           &SA1,&SAS1,&SA2,&SAS2               Определение дейст-
                                                    вительных переменных
```

Б – Ввод исходных данных из файла-меню в модель

```
GETLIST   FILE=ROBO,(&M1)          Выключатель рабочего места 1
.
.
.
GETLIST   FILE=ROBO,(&RASG)        Ср. продолжительность разгрузки
                                   робокара
GETLIST   FILE=ROBO,(&RASGR)       Отклонение от средней
                                   продолжительности
GETLIST   FILE=ROBO,(&SHIFT)       Продолжительность моделиро-
                                   вания
```

В – моделирование технологических операций с вводом команд файла управления анимацией

```
**************************************************************
*                  Моделирование рабочего места 1
**************************************************************
      GENERATE   &SA1,&SAS1,,,,1PL,1PH    Поступление заявки с
                                          рабочего места 1
      TEST NE    &M1,0,DES                Проверка включения
                                          рабочего места 1
      ASSIGN     1,&RES1/&SCOR,PL         Время движения до
                                          рабочего места 1
      ASSIGN     1,1,PH                   Принадлежность заявки
                                          1 рабочему месту
      BPUTPIC    FILE=ATF,LINES=6,AC1,&SA1,&SAS1,&RES1,PL1
                                          Запись в файл .atf в 6 линий
                                    следующих команд:
time *.*              Текущее время
create m m1           Создать динамический объект – рабочее место 1
place m1 -30 38       Поместить объект в координатное пространство Proof
                      Animation
set m1 color red      Окрасить рабочее место 1 в красный цвет
write z1 *  *         Вывести на экран математическое ожидание и
                      дисперсию поступления заявки
write r1 L=*m         Вывести на экран расстояние от склада до
                    рабочего места 1
write re1 t=*min      Вывести на экран продолжительность движения ро-
                      бокара до рабочего места 1
      TRANSFER,ROB          Переход заявки на обслуживание к робокару
      .
.
.

**************************************************************
*                  Моделирование работы робокара
**************************************************************

ROB   QUEUE     HUK             Ожидание освобождения робокара
      SEIZE     CAR             Занятие робокара
      DEPART    HUK             Конец ожидания
      ADVANCE RVNORM(1,&POG,&POGR)  Загрузка робокара
```

Г – Вывод результатов моделирования на экран

```
        BPUTPIC         FILE=ATF,LINES=6,AC1,N(BOX),FT(CAR),_
                        FR(CAR)/10,QA(HUK),QT(HUK)
        time *.*        Текущее время
        write b *       Количество обслуженных заявок
        write ft *.**   Среднее время обслуживания заявки
        write fr *.**   Загрузка робокара
        write qa *.**   Средний размер очереди на обслуживание
        write qt *.**   Среднее время ожидания заявки в очереди
DES     TERMINATE       Заявка выполнена
```

Этап 3. Отображение динамики технологического процесса при помощи языка компьютерной анимации Proof Animation.

Динамику работы робокара отобразим на компьютерной мнемосхеме технологического процесса при помощи языка компьютерной анимации Proof Animation. Анимация состоит из статических элементов и динамических объектов, которые накладываются на статический фон. В нашем случае статическими элементами являются траектории движения робокара, склад и надписи, а динамическим объектом – робокар.

Для рисования статических элементов используем меню Mode и режим Draw Mode (рис. 50).

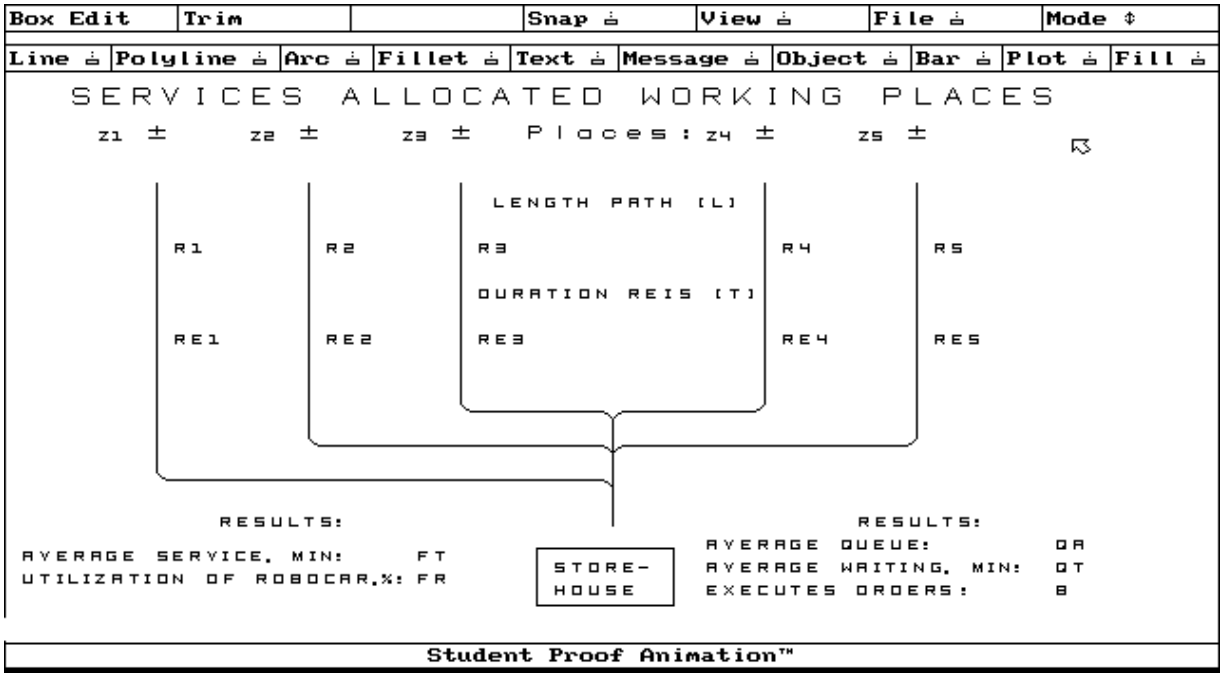


Рис. 50. Статические элементы анимации

Рисование динамического объекта осуществляем в режиме Class Mode (рис. 51). Команды управления динамическими элементами для Proof Animation: TIME, CREATE, PLACE AT, SET COLOR, WRITE и др. – заранее введены в программу на языке GPSS/H (см. выше). В совокупности эти команды при работе программы GPSS/H образуют файл управления анимацией (.atf-файл).

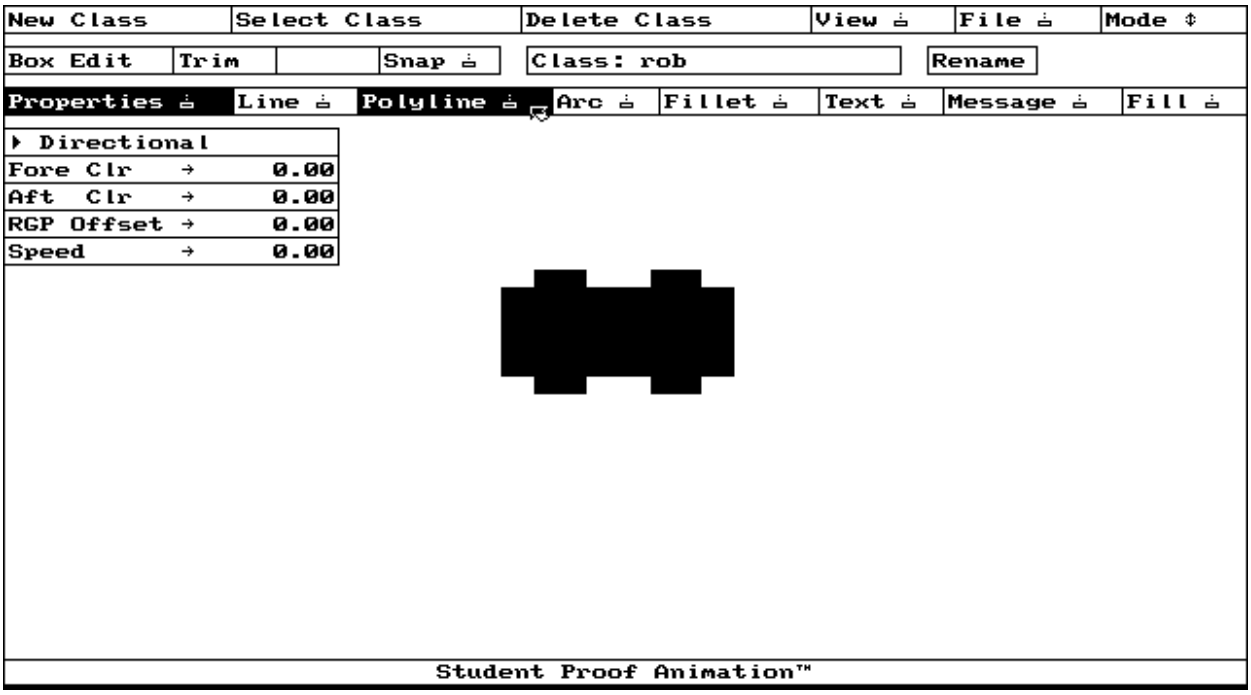


Рис. 51. Динамический объект анимации

Один из кадров анимации процесса обслуживания рабочих мест робокаром, полученный после имитационного моделирования, показан на рис. 52.

При подаче заявки соответствующее рабочее место изменяет цвет, а после обслуживания возвращается к первоначальному цвету. Перемещение робокара в пространстве отображается движением его графического аналога на мнемосхеме с соответствующей скоростью. Занятость робокара также отображается сменой цвета. На экран выводится: количество обслуженных заявок; средняя продолжительность обслуживания заявки; загрузка робокара и другие параметры.

В верхней строке экрана указывается: текущее время имитации (Time), скорость анимации (Speed), ускорение (Faster) или замедление (Slower) анимации, остановка изображения (Pause), пуск дви-

жущегося изображения (Go), выбор размера, расположения и участка изображения (View), выбор файлов анимации процесса (File), выбор режимов анимации (Mode).

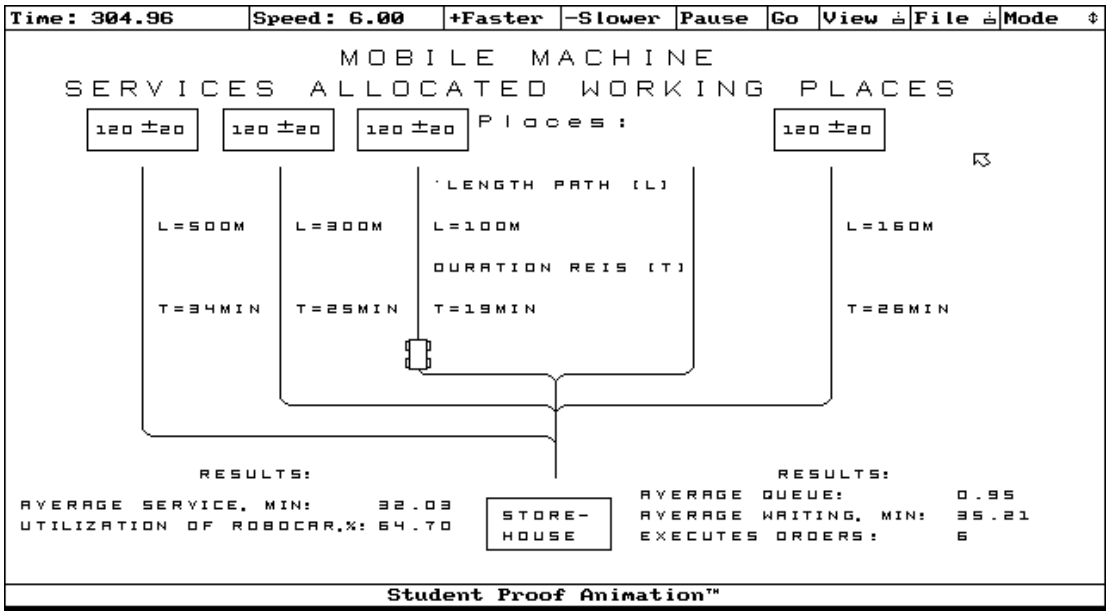


Рис. 52. Кадр анимации процесса обслуживания рабочих мест робокаром

Этап 4. Оценка характеристик технологического процесса на разработанных моделях компьютерной имитации и анимации.

В имитационных экспериментах вводим исходные данные в файл-меню, запускаем GPSS/H-модель и анализируем результаты по сформированной анимации процесса.

В результате проведения имитационных экспериментов установлено, что за 8-часовой рабочий день обслужено 10 заявок. При этом средняя продолжительность обслуживания заявки составляла 33,93 мин, загрузка робокара – 76,40 %. Средний размер очереди заявок на обслуживание из всех рабочих мест равнялся 2,78. При этом среднее время ожидания заявки в очереди было 82,30 мин.

Для исследования изменения загрузки робокара при изменении числа рабочих мест изменяли число источников заявок (от 1 до 5) и оценивали степень использования робокара для каждого эксперимента.

Подобным образом оценивалось изменение среднего размера очереди заявок на обслуживание от числа рабочих мест при различной скорости движения робокара.

По результатам имитационных экспериментов, выведенным на экран, были построены зависимость степени использования робокара от числа рабочих мест и зависимость среднего размера очереди заявок на обслуживание от числа рабочих мест и скорости робокара (рис. 53, 54).

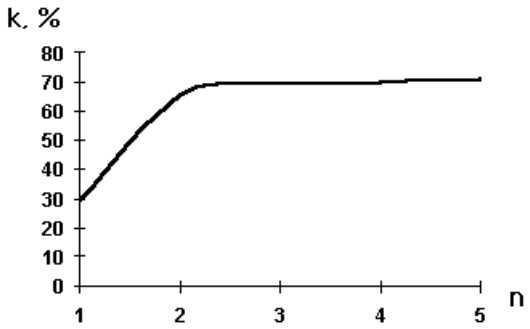


Рис. 53. Зависимость степени использования робокара *k* от числа рабочих мест *n*

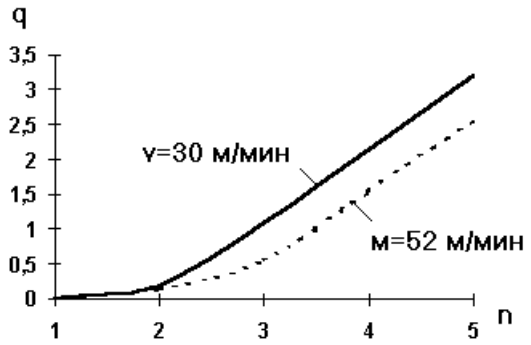


Рис. 54. Зависимость среднего размера очереди заявок на обслуживание *q* от числа рабочих мест *n* и скорости движения робокара *v* (– *v* = 30 м/мин; - - *v* = 52 м/мин)

Установлено, что увеличение числа рабочих мест от 1 до 2 приводит к повышению степени использования робокара на 36 %. Увеличение числа рабочих мест от 2 до 5 несущественно влияет на загрузку робокара. Изменение составляет 2 %.

Средний размер очереди заявок на обслуживание практически не изменяется при числе рабочих мест 1-2. При этом изменение скорости робокара не оказывает никакого влияния на средний размер очереди заявок на обслуживание. При числе рабочих мест от 2 до 5 увеличение скорости робокара уменьшает средний размер очереди в среднем на 26 %.

ЗАКЛЮЧЕНИЕ

Математическое моделирование используется в случае, если технологическая система не поддается физическому эксперименту. Во многих случаях невозможно получить аналитические зависимости, отображающие поведение и взаимосвязь элементов технологической системы. Особенно трудно учесть действие случайных факторов и динамику функционирования объекта. Поэтому используют имитационное моделирование. Имитационное моделирование похоже на физические эксперименты, но эти эксперименты проводятся не на физическом объекте, а на его компьютерной модели. Сущность имитационного моделирования состоит в том, что над моделью проводят эксперименты типа «Что, если…?». Изменяя исходные показатели в модели, анализируют результаты экспериментов. Имитационная модель реализуется на ЭВМ при помощи языков программирования. Альтернативой является использование методов, основанных на специализированных языках и системах имитационного моделирования. Специализированные языки содержат, как правило, написанные на универсальном языке блоки – отдельные динамические модели. Поведение системы имитируется как смена ее состояний. Процесс отображается не системой уравнений, а взаимодействием отдельных динамических моделей во времени и пространстве. Поведение системы описывается от события к событию, означающих начало или окончание технологической операции.

В настоящем учебном пособии рассматривалось применение одного из наиболее распространенных специализированных языков имитационного моделирования – GPSS, предложенного фирмой IBM в 1962 г., и программы компьютерной анимации (CIMAN – CINEMA, GPSS/H – Proof Animation), позволяющих отображать динамику процесса на экране компьютера в соответствии с работой имитационной модели и выводить результаты на монитор.

ПРИЛОЖЕНИЯ

Приложение 1

Файл-меню модели процесса обслуживания рабочих мест робокаром

```
1          *Рабочее место 1 (включено 1; выключено 0)
1          *Рабочее место 2 (включено 1; выключено 0)
1          *Рабочее место 3 (включено 1; выключено 0)
1          *Рабочее место 4 (включено 1; выключено 0)
1          *Рабочее место 5 (включено 1; выключено 0)
120        *Среднее время поступления заявки с рабочего места 1
20         *Отклонение от среднего времени
120        *Среднее время поступления заявки с рабочего места 2
20         *Отклонение от среднего времени
120        *Среднее время поступления заявки с рабочего места 3
20         *Отклонение от среднего времени
120        *Среднее время поступления заявки с рабочего места 4
20         *Отклонение от среднего времени
120        *Среднее время поступления заявки с рабочего места 5
20         *Отклонение от среднего времени
500        *Расстояние до рабочего места 1
300        *Расстояние до рабочего места 2
100        *Расстояние до рабочего места 3
200        *Расстояние до рабочего места 4
160        *Расстояние до рабочего места 5
30         *Скорость робокара, м/мин
8          *Средняя продолжительность загрузки робокара, мин
2          *Отклонение от среднего времени, мин
8          *Продолжительность разгрузки робокара, мин
2          *Отклонение от среднего времени, мин
480        *Продолжительность моделирования, мин
**********************************************************
```

Распечатка GHSS/H-программы с вводом команд управления анимацией

```
**********************************************************************
*          Модель процесса обслуживания рабочих мест робокаром
**********************************************************************
*          Создание .atf-файла и определение переменных
**********************************************************************
SIMULATE                                        Начало модели
                                                рования
ATF   FILEDEF    'ROBO.ATF'                     Создание .atf
                                                файла
```

Приложение 2

```
INTEGER   &M1,&M2,&M3,&M4,&M5,&SHIFT        Целые пере-
                                             менные
REAL          &SA1,&SAS1,&SA2,&SAS2          Действитель
                                             ные перемен
                                             ные
REAL          &SA3,&SAS3,&SA4,&SAS4,&SA5,&SAS5
REAL          &RES1,&RES2,&RES3,&RES4,&RES5
REAL          &SCOR,&POG,&POGR,&RASG,&RASGR
************************************************************
      Ввод исходных данных из файла-меню
************************************************************
GETLIST   FILE=ROBO,(&M1)     Выключатель рабочего места 1
GETLIST   FILE=ROBO,(&M2)     Выключатель рабочего места 2
GETLIST   FILE=ROBO,(&M3)     Выключатель рабочего места 3
GETLIST   FILE=ROBO,(&M4)     Выключатель рабочего места 4
GETLIST   FILE=ROBO,(&M5)     Выключатель рабочего места 5
GETLIST   FILE=ROBO,(&SA1)    Среднее время поступления
                              заявки с рабочего места 1
GETLIST   FILE=ROBO,(&SAS1)   Отклонение от среднего времени
GETLIST   FILE=ROBO,(&SA2)    Среднее время поступления
                              заявки с рабочего места 2
GETLIST   FILE=ROBO,(&SAS2)   Отклонение от среднего времени
GETLIST   FILE=ROBO,(&SA3)    Среднее время поступления
                              заявки с рабочего места 3
GETLIST   FILE=ROBO,(&SAS3)   Отклонение от среднего времени
GETLIST   FILE=ROBO,(&SA4)    Среднее время поступления
                              заявки с рабочего места 4
GETLIST   FILE=ROBO,(&SAS4)   Отклонение от среднего времени
GETLIST   FILE=ROBO,(&SA5)    Среднее время поступления
                              заявки с рабочего места 5
GETLIST   FILE=ROBO,(&SAS5)   Отклонение от среднего времени
GETLIST   FILE=ROBO,(&RES1)   Расстояние от склада до рабочего
                              места 1
GETLIST   FILE=ROBO,(&RES2)   Расстояние от склада до рабочего
                              места 2
GETLIST   FILE=ROBO,(&RES3)   Расстояние от склада до рабочего
                              места 3
GETLIST   FILE=ROBO,(&RES4)   Расстояние от склада до рабочего
                              места 4
GETLIST   FILE=ROBO,(&RES5)   Расстояние от склада до рабочего
                              места 5
GETLIST   FILE=ROBO,(&SCOR)   Скорость движения робокара
```

Продолжение прил. 2

```
GETLIST   FILE=ROBO,(&POG)    Средняя продолжительность
                              загрузки робокара
GETLIST   FILE=ROBO,(&POGR)   Отклонение от среднего времени
GETLIST   FILE=ROBO,(&RASG)   Средняя продолжительность
                              загрузки робокара
GETLIST   FILE=ROBO,(&RASGR)Отклонение от среднего времени
GETLIST   FILE=ROBO,(&SHIFT)  Продолжительность
                              моделирования
PUTPIC    FILE=ATF,LINES=8,AC1
time *.*
create m m1
create m m2
create m m3
create m m4
create m m5
create rob rob
place rob 0 6
************************************************************
*                    Рабочее место 1
************************************************************
GENERATE  &SA1,&SAS1,,,,1PL,1PH   Поступление заявки с
                                  рабочего места 1
TEST NE   &M1,0,DES               Проверка включения
                                  рабочего места 1
ASSIGN    1,&RES1/&SCOR,PL        Время движения до
                                  рабочего места 1
ASSIGN    1,1,PH                  Принадлежность заявки
                                  1-му рабочему месту
BPUTPIC   FILE=ATF,LINES=6,AC1,&SA1,&SAS1,&RES1,PL1
TIME *.*
set m1 color red
place m1 -30 38
write z1 *  *
write r1 L=*m
write re1 t=*min
TRANSFER,ROB                      Переход к робокару
************************************************************
*                    Рабочее место 2
************************************************************
GENERATE  &SA2,&SAS2,,,,1PL,1PH   Поступление заявки с
                                  рабочего места 2
TEST NE   &M2,0,DES               Проверка включения
```

Продолжение прил. 2

```
                                        рабочего места 2
ASSIGN     1,&RES2/&SCOR,PL             Время движения до
                                        рабочего места 2
ASSIGN     1,2,PH                       Принадлежность заявки
                                        2-му рабочему месту
BPUTPIC    FILE=ATF,LINES=6,AC1,&SA2,&SAS2,&RES2,PL1
time *.*
set m2 color red
place m2 -20 38
write z2 *  *
write r2 L=*m
write re2 t=*min
TRANSFER,ROB                            Переход к робокару
************************************************************
*                    Рабочее место 3
************************************************************
GENERATE   &SA3,&SAS3,,,,1PL,1PH        Поступление заявки с
                                        рабочего места 3
TEST NE    &M3,0,DES                    Проверка включения
                                        рабочего места 3
ASSIGN     1,&RES3/&SCOR,PL             Время движения до
                                        рабочего места 3
ASSIGN     1,3,PH                       Принадлежность заявки
                                        рабочему месту 3
BPUTPIC    FILE=ATF,LINES=6,AC1,&SA3,&SAS3,&RES3,PL1
TIME *.*
set m3 color red
place m3 -10 38
write z3 *  *
write r3 L=*m
write re3 t=*min
TRANSFER,ROB                            Переход к робокару
************************************************************
*                    Рабочее место 4
************************************************************
GENERATE   &SA4,&SAS4,,,,1PL,1PH        Поступление заявки с
                                        рабочего места 4
TEST NE    &M4,0,DES                    Проверка включения
                                        рабочего места 4
ASSIGN     1,&RES4/&SCOR,PL             Время движения до
                                        рабочего места 4
ASSIGN     1,4,PH                       Принадлежность заявки
```

Продолжение прил. 2

```
                                        рабочему месту 4
BPUTPIC   FILE=ATF,LINES=6,AC1,&SA4,&SAS4,&RES4,PL1
TIME *.*
set m4 color red
place m4 10 38
write z4 *  *
write r4 L=*m
write re4 t=*min
TRANSFER,ROB                            Переход к робокару
************************************************************
      Рабочее место 5
************************************************************
GENERATE  &SA5,&SAS5,,,,1PL,1PH         Поступление заявки с
                                        рабочего места 5
TEST NE   &M5,0,DES                     Проверка включения
                                        рабочего места 5
ASSIGN    1,&RES5/&SCOR,PL              Время движения до
                                        рабочего места 5
ASSIGN    1,5,PH                        Принадлежность заявки
                                        рабочему месту 5
BPUTPIC   FILE=ATF,LINES=6,AC1,&SA5,&SAS5,&RES5,PL1
TIME *.*
set m5 color red
place m5 20 38
write z5 *  *
write r5 L=*m
write re5 t=*min
************************************************************
*                  Моделирование работы робокара
************************************************************
ROB  QUEUE     HUK                      Ожидание робокара
SEIZE      CAR                          Занятие робокара
DEPART     HUK                          Конец ожидания
ADVANCE RVNORM(1,&POG,&POGR)            Загрузка робокара
BPUTPIC   FILE=ATF,LINES=4,AC1,PH1,PL1/2,PH1
time *.*
set rob color red
set path p* travel *.*
place rob on p*
ADVANCE PL1                             Движение робокара к
                                        рабочему месту
ADVANCE RVNORM(1,&RASG,&RASGR)      Разгрузка робокара
```

Продолжение прил. 2

```
BPUTPIC    FILE=ATF,LINES=5,AC1,PH1,PH1,PL1/2,PH1
time *.*
set rob color green
set m* color green
set path p1* travel *.*
place rob on p1*
ADVANCE PL1                                  Движение робокара к
                                              складу
BOX   RELEASE          CAR                   Освобождение робокара
BPUTPIC    FILE=ATF,LINES=6,AC1,N(BOX),FT(CAR),_
           FR(CAR)/10,QA(HUK),QT(HUK)
time *.*
write b *
write ft *.**
write fr *.**
write qa *.**
write qt *.**
DES   TERMINATE                                Заявка выполнена
************************************************************
*                  Продолжительность моделирования
************************************************************
GENERATE       &SHIFT          Появление транзакта-таймера
TERMINATE      1               Удаление транзакта-таймера
START          1
PUTPIC         FILE=ATF,LINES=2,AC1
time *.*
END
      END
```

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Советов, Б. Я. Моделирование систем : учеб. для вузов / Б. Я. Советов, С. А. Яковлев. – 3-е изд., перераб. и доп. – М. : Высш. шк., 2001. – 343 с.

2. Шеннон, Р. Имитационное моделирование систем – искусство и наука. – М. : Мир, 1978. – 420 с.

3. Кузин, Л. Т. Основы кибернетики : учеб. пособие для студентов вузов / Л. Т. Кузин. – М. : Энергия, 1979. – 584 с.

4. Шрайбер, Т. Д. Моделирование на GPSS. – М. : Машиностроение, 1980. – 593 с.

5. Основы моделирования на GPSS/PC : метод. указания для слушателей ФПКП по моделированию систем и сетей связи на GPSS/PC / сост. : Л. А. Воробейчиков, Г. К. Сосновиков. – М. : МТУСиИ, 1993. – 58 с.

6. Разработка САПР : методические указания по лабораторной работе «Имитационное моделирование систем массового обслуживания» / сост. : И. П. Норенков, Е. В. Федорук. – М. : МГТУ им. Н. Э. Баумана, 1994. – 67 с.

7. Компьютерная имитация и анимация : метод. указания по выполнению курсовой работы / сост. : В. В. Зиновьев. – Кемерово : КузГТУ, 2001. – 12 с.

8. Компьютерная имитация и анимация : программа, методические указания и контрольные задания для студентов заочной формы обучения специальности 210200 «Автоматизация технологических процессов и производств». – Кемерово : КузГТУ, 2002. – 32 с.

9. Построение анимационных моделей в Proof Animation : методические указания к лабораторной работе / сост. : А. В. Протодьяконов, А. А. Будников, В. В. Зиновьев. – Кемерово : КузГТУ, 2001. – 16 с.

Зиновьев Василий Валентинович
Стародубов Алексей Николаевич

Моделирование систем при помощи компьютерной имитации и анимации

Учебное пособие

Редактор Савина З. М.

Подписано в печать 19.10.2010. Формат 60×84/16
Бумага офсетная. Отпечатано на ризографе. Уч.-изд. л. 7,00
Тираж 150 экз. Заказ

Кузбасский государственный технический университет
650000, Кемерово, ул. Весенняя, 28

Типография Кузбасского государственного технического университета
650000, Кемерово, ул. Д. Бедного, 4А