**Моделирование**

1. **Понятие имитационного моделирования. Формальная модель объекта. Типовые математические схемы моделирования**

**Модель** – любое мысленное формальное физическое представление объекта окружающего мира обеспечивающее изучение определённых его св-в. **Моделирование** – это замещение одного объекта (оригинала) другим (моделью) и фиксация или изучение свойств оригинала путем исследования свойств модели. **Система** – группа объектов, объединенных какой-либо формой взаимодействия с целью выполнения опред. задач. Система определяется совокупностью параметров и характеристик. Множество параметров системы отражает ее внутреннее содержание – структуру и принципы функционирования. Характеристики системы – это ее внешние свойства, которые важны при взаимодействии с другими системами. Характеристики системы находятся в функциональной зависимости от ее параметров. **Динамические системы** – сост-е с теч времени мен-ся непрерывно.

Методы мод-я: **1) аналитические** – позвол получить хар-ки с-мы как некот ф-ции параметров ее функ-вания. Аналит модель – с-ма дифф ур-ий. **2) имитационные** - представ-е динамич поведения с-мы посредством продвиж-я ее от одного сост-я к др в соотв-вии с опред-ми имитац-ми правилами. Шеннон: **Критерии прим-я** им-го мод-я: 1) не сущ-ет законченной матем постановки задачи 2) аналитич методы имеются, но настолько сложны и трудоемки, что рациональнее исп-ать им мод-е 3) кроме оценки опред-х парам-ов необх-мо наблюдение за ходом процесса.

**Проблемы применения им мод**: 1) нахождение компромисса м\д сложностью и упрощением модели 2) искусственном воспроизводство случ воздействий окр среды 3) оценка адекватности модели.

**Математические модели систем**

На базе мат модели происх анализ хар-ик с-мы, при комп-ом мод-ии на основе мат модели созд-ся алгоритм программ д\получ-я инф-ии о поведении с-мы, **формальное описание объекта.**

В общем сл мат модель любой динамической с-мы м\представить в виде: 

**x**- совок-сть входных воздействий, **h**- совок внутр-х параметров с-мы, **y**- совок выходных хар-ик, **t**– время, **F** - закон ф-ния. Процесс функц-я с-мы м\рассм-ать как послед-ную смену сост-ий

:

,

где z- совок начальных состояний.

Т.о, **общую мат модель** с-мы м\представить: .

При построении мат м\выделить осн подходы: **1)** непрерывно-детерминиров – исп-ет в кач-ве мат моделей с-мы дифф ур-ий (сост-е с-мы изм-ся с теч вр по непрерыв-му строго опред закону: колебания маятника) **2)** дискретно-детерм – с-ма мен сост-я по опред закону, h – конечное число сост-ий. Реализ-ся с пом мат аппарата теории автоматов (д\упр-я быт тех) **3)** дискретно-стохастический – вероятностный автомат (д\анализа надежности ИС) **4)** непрерывно-стох – с-ма мен сост-я с теч вр по случ закону (СМО) **5)** сетевой (сети Петри) 6) агрегативный подход(универсальный).

1. **Непрерывно-стохастические модели (Q-схемы).**

С-ма меняет свои сост-я с теч вр по случ закону (СМО).

Типовой мат схемой мод-я таких с-м явл Q-схемы. В обслуживании м\выделить 2 составляющие: ожидание обсл-я и обсл-е, а в любой СМО м\выделить элементарный прибор. В нем выделяют накопитель, некоторой емкостью; канал; потоки событий: поток заявок на обслуж-е wi, хар-щийся моментами поступления и атрибутами заявок, и поток обслуж-я ui, хар-щийся моментами начала и окончания обслуж-я заявок. Рис.

Различают: **поток однородных** событий (хар-ся только моментами поступления этих событий и задается послед-стью {tn }={0<= t1<= t2 ... <= tn <=...}, где tn - момент наступления n-го события. **Потоком неоднородных** событий (последовательность {tn,fn }, где fn - набор признаков события).

**Допущения**

**Ординарный поток** P>1(t, ) << P1(t, ). Для любого интервала  верно следующее:

P0(t, )+ P1(t, )+ P>1(t, )=1

**Стационарный поток** - поток, для кот вер-ть появления того или иного числа событий зав от длины интервала и не зав от того, где на оси времени взят этот интервал.

Среднее число событий, наступающих на интервале врем :

0·P0(t, )+1·P1(t, ) = P1(t, )

Тогда ср число событий орд-го потока в ед времени (**интенсивность потока**):



Для стационар потока a(t)=a= const/

Процесс ф-ния прибора м\представить как процесс изм-я сост-ий его элем-ов во времени z(t). Вектор сост-ий д\прибора имеет вид , где zн – сост-е накопителя, zк – сост-е канала.

Неоднородность заявок, отражающая процесс в реальной с-ме, учитывается с пом классов приоритетов. Различа статические (назначаются заранее) и динамические (возникают при мод-ии). **Относительный приоритет** - заявка с более выс приоритетом, поступившая в Н, ожидает окончания обслуж-я предшествующей заявки и только после этого занимает канал. **Абсолютный приорит**ет - заявка с более выс приоритетом, поступившая в Н, прерывает обслуж-е заявки с более низким приоритетом и занимает канал.

1. **Сущность метода статистического моделирования. Примеры использования.**

**Основа** - метод статистических испытаний Монте-Карло, кот базир-ся на исп-ии случ чисел, т.е возможных знач-ий некот случ вел-ны с заданным распред-ем вер-тей. **Сущность** метода: построение для процесса функц-я исследуемой с-мы некот-го моделирующего алг-ма, имитирующего поведение и взаимод-е эл-ов с-мы с учетом случ-х входных воздействий и воздействий внеш среды и реализация этого алг-ма на ЭВМ.

**Обл прим-я**: 1) изучение стохаст-х с-м; 2) решение детерминированных задач.

Рез-тат статист-го мод-я - серия частных знач-ий искомых величин или ф-ий, их статистическая обраб-ка. Если кол-во реализаций N → ∞, рез-ты устойчивы и достаточно точны.

**Пример1** (детерм). S фигуры.





1. **Генерация случайных чисел. Генерация последовательностей псевдослучайных чисел. Требования к генератору псевдослучайных чисел. Улучшение качества последовательностей.**

3 способа генерации случайных чисел:

**1)Аппаратный** - в осн лежит к-л физич эффект. «-» не гарант-ет кач-во послед-сти сл чисел во время мод-я; нельзя получать одинак послед-сти.

**2)Табличный** - случ числа оформлены в виде табл. «-» запас чисел ограничен, выч ресурсы исп-ся неэффект.

**3)Программный** (алгоритмический) – случ числа формир-ся с пом спец прог. Генерир-е некот базовых процессов и их последующее преобраз-е. Чаще всего в кач-ве базовой послед-ти исп-ют независимые случ величины, равномерно распред-ные на (0,1).**Рис**

Для получения случ чисел на ЭВМ исп-ся алг-мы, поэт такие последоввт-сти, являющиеся по сути детерминированными, назыв **псевдослучайными**.

ЭВМ оперирует n-разрядными числами, поэтому вместо непрерывной совок-ти равномерных случ чисел интервала (0,1) исп-ют дискретную послед-ть 2n случ чисел того же интервала - закон распред-я такой дискретной послед-ти назыв квазиравномерным распред-ем.

**Требования к идеальному генератору:**

1)Послед-ть д\состоять из квазиравномерно распред-х чисел 2)Числа д\б независимы 3)Послед-ти случ чисел д\б воспроизводимыми 4)Послед-ти д\иметь неповторяющиеся числа 5)Минимал затраты вычислительных ресурсов.

Наиб прим-е на ЭВМ д\генерации послед-тей псевдослучайных чисел находят алг-мы вида: xi+1=Ф(xi), представ собой реккурентные соотношения 1го порядка.

«-» наличие коррелляции м\д числами послед-ти, иногда случ-сть отсутствует.

Прим-ся конгруэнтные процедуры генерации псевдослуч-х послед-тей.

2 целых числа конгруэнтны (сравнимы). Осн на формуле: 

где  - неотрицат целые числа.

**Методы улучшения качества:**

1)Исп-е рекуррентных формул большего порядка r:



2)Метод возмущающих ф-ций:



1. **Моделирование случайных воздействий (моделирование случайных событий). Моделирование дискретной случайной величины.**

**1. Необх.реализовать случ.событие А, наступающее с заданной вероятностью p**: Определим А как событие, состоящ.в том, что выбран.знач-е xi равномерно распределенной на интервале (0,1) случ.величины удовлетворяет неравенству: xi ≤ p. Тогда вер-сть события А будет P(A) = ∫[0..p](dx) = p; Противоположн.событие состоит в том, что xi >p, его вер-сть равна 1-р.

**2. Рассм. группу событий**: Пусть А1, А2 ,..., Аs - полная группа событий, наступающ.с вер-стями p1, p2 ,..., ps. Определим событие Аm как событие, состоящ. в том, что выбран. Знач-е xi случ. величины удовлетв. Нер-ву: lm-1 < xi ≤ lm; где lr = ∑[i=1..r](pi);

Процедура моделир-я испытаний в этом случае состоит в последоват. сравнении случ.чисел xi со значениями lr. Если условие выполн-ся, исходом испытания оказыв-ся событие Аm .

**3. Рассм. независимые события А и В с вер-стями наступления рА и рВ:** Возможными исходами совмест.испытаний в этом случае будут события АВ, ]AB, A]B,]A]B с вер-стями рА∙рВ, (1-рА)∙рВ, рА∙(1-рВ), (1-рА)∙(1-рВ). Для моделир-я совместн. испытаний можно исп-ть два вар-та процедуры: \* Последоват.е выполнение процедуры.\* Определ-е одного из исходов АВ, ]AB, A]B,]A]B по жребию с соответствующими вер-стями.

**4. События А и В явл-ся зависимыми и наступают с вер-стями pА и pВ :** Обозначим через pА(В) условную вер-сть наступления события В при условии, что событие А произошло. Алгоритм модели может быть:

[Пуск] → [Генерация] → <[xi < pA]> → (1+ or 2-) →

(1+) → [KA = KA+1] → [Генерация] → (1a+ or 1b-) →

(1a+) → [KAB=KAB+1] → [выход]

(1b-) → [KANB=KANB+1] → [выход]

(2-) → [KNA=KNA+1] → [генерация] → <[xi < pNA(B)]> → (2a+ or 2b-) →

(2a+) → [KNAB=KNAB+1] → [выход]

(2b-) → [KNANB=KNANB+1] → [выход];

**Моделир-е дискретных случ.величин:**

Дискрет. Случ.вел-на Y принимает знач-я y1 ≤ y2 ≤ … ≤ yj ≤ … с вер-стями p1, p2, …, pj,… составляющими дифференц. Распредел-е вер-стей.

Для получения дискрет. Случ. величин можно воспольз-ся методом обратных функций: если X – равномерно распред. на интервале (0,1) случ. величина, то искомую случ. величину получают при выполнении действий:

Если x1 < p1 , то Y= y1 , иначе,

Если x1 < p1 + p2, то Y= y2 , иначе, ……..

Если x1 < ∑[j=1..m]Pj; то Y= ym;

1. **Моделирование непрерывных случайных величин (метод обратных преобразований, показательный закон, треугольный закон распределения).**

Для получения НСВ (непрерывных случайных величин) с заданным законом распределения можно использовать **метод обратной ф-ии**. Если СВ (случайная величина) Y имеет плотность распределения f(y), то распределение СВ: F(y)=∫[0..y]f(y)dy; явл-ся равномерным на интервале (0,1). Чтобы получить число, принадлежащее последоват-ти СЧ (случайных чисел) {yi}, имеющих ф-ю плотности f(y), надо разрешить относительно **yi** ур-е: xi=∫[0..y]f(y)dy;

где xi - число, принадлежащее последовательности СЧ равномерно распределенных на интервале от (0,1).

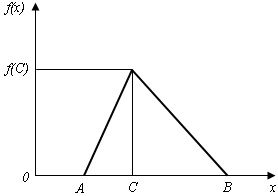
**Показательный закон распределения**: Необходимо получить случайные числа с показательным законом распределения (напр-р, интервалов времени м\ду поступлениями заявок на обслуживание): f(t)=λ∙e-λt; xi=∫[0..t](λ∙e-λt)dt; t=-(1/λ)∙ln(xi);

Этот способ получения случ-ых чисел с заданным законом распределения имеет ограниченную сферу применения, т.к. для многих законов распределения, встречающихся в практических задачах моделирования, интеграл не берется, => приходится прибегать к численным методам реш-я, что увеличивает затраты вычислительных ресурсов на получение каждого числа; Поэтому на практике пользуются приближенными способами преобраз-я СЧ, кот-е делят на:

а) универсальные способы, с пом-ю кот-х можно получать СЧ с законом распределения любого вида; б) неуниверсальные способы, пригодные д\получения случ-ых чисел с конкретным законом распределения.

**Треугольное распределение:** применяется когда о случайной величине ничего неизвестно, кроме наиболее вероятного значения и диапазона возможных значений этой случайной величины

Для получения последовательностей СЧ, подчиненных треугол-му распределению исп-ся метод обратных функций.



1. **Моделирование непрерывных случайных величин (универсальный метод (кусочная аппроксимация), нормальный закон распределения).**

**Показат.з-н распредел-я**: Необх.получить случ. числа с показат. З-ном распредел-я (например, интервалов времени между поступлениями заявок на обслуживание): f(t)=λ∙e-λt; xi=∫[0..t](λ∙e-λt)dt; t=-(1/λ)∙ln(xi);

Этот способ получения случ. чисел с заданным з-ном распредел.имеет огранич.сферу применения, т.к. для многих з-нов распредел-я, встречающ. в задачах моделир-я, интеграл не берется, т.е. приходится прибегать к числен. методам решения, что увелич. затраты вычислит. ресурсов на получение каждого числа; на практике пользуются приближен.способами преобразов-я случ. чисел, кот.можно классифицировать:

а) универс. способы, с пом. которых можно получать случ. числа с з-ном распредел-я любого вида; б) неуниверсальн. способы, пригодные для получения случ. чисел с конкретным з-ном распредел-я.

Рассм.приближ.универс.способ получения случ. чисел, основанный на кусочной аппроксимации ф-ии плотности.

Пусть треб-ся получить послед-сть случ.чисел {yj} с ф-ей плотности fη(y) , знач-я кот. лежат в интервале (a,b). Разобьем интерв. (a,b) на m интервалов, и будем считать fη(y) на каждом интерв.постоянной. Разбивать необх-мо так, чтобы вер-сть попадания случ.вел-ны в любой интервал (ak, ak+1) была постоянной, т.е.: ∫[ak…ak+1]fn(y)dy;

алгоритм этого способа получения случ. чисел сводится к выполн-ю след. действий:

1) Генерир-ся случ.равномерно распредел. число xi из интервала (0,1);

2) с пом. этого числа выбир-ся интерв. (ak, ak+1);

3) генерир-ся число xi+1 ;

4) вычисл-ся случ. число yj =ak+ xi+1(ak+1-ak) с требуемым з-ном распредел-я.

Вид норм.распред-я

В современ. системах имитац.моделир-я обычно исп-ся не менее двух програм. датчи­ков случ.величин, распред.по норм. Закону.



1. **Управление модельным временем (принцип Δt и принцип δz).**

При создании модели важным явл-ся реализация 2-х функций: 1) корректировка временной координаты состояния сист-ы ("продвижение" времени, организация "часов"); 2) обеспечение согласованности различных блоков и событий в сист-е (синхронизация во времени, координация с др. блоками). Т.о., функционирование модели должно проте­кать в искусственном (не в реальном и не в машинном) времени, обеспечивая появление событий в требуемом логикой работы ис­следуемой системы порядке и с надлежащими временными интерва­лами м\ду ними.

1) **принцип Δt (квантования времени).** Состояние с-мы хар-ся вектором сост-ий Z(t). Организуем счетчик систем-го времени, кот в начальный момент показ время t0. Прибавим интервал врем Δt, тогда счетчик будет показывать t1= t0 +Δt. Вычислим знач-я Z(t0 +Δt), затем перейдем к мом врем t2= t1 +Δt и т.д. Если шаг Δt достаточно мал, то таким путем можно получать приближенные знач-я Z.

**2) принцип δz.** При рассм процессов функц-ния некот сист-м можно обнаружить, что для них характерны 2 типа состоя­ний: 1) особые, присущие процессу ф-ия с-мы то­лько в некоторые мом-ы времени 2) неособые, в кот-ых процесс находится всё остальное время. Особые состояния хар-ны еще и тем, что функции сост-ий Z(t) в эти мом врем изм-ся скач­ком, а м\ду особыми сост-ми изменение координат Z(t) происходит плавно и непрерывно или не происх вообще. Для описанного типа сист-м м\б построены моделирующие алг-мы по «принципу особых состо­яний». Обозначим скачкообразное изм-ие сост-я z как δz. Хар-ки процесса функц-я таких сист-м оцениваются по инф-ии об особых сост-ях, а неособые состояния при мод-ии не рассм-ся.

1. **Механизм протяжки модельного времени.**

1 Время реальной системы – время в которой живет и функционирует реальная система.

2 Модельное время – искусственное время в кот-ом живет модель, или время которое явл-ся имитационным временем реальной сист-ы.

3 Реальное время – время в кот-ом живет исследователь и компьютер; время необходимое для моделирования.

В моделирующих сист-ах выделяют 2 списка событий: cписок текущих событий (СТС) и список будущих событий (СБС). Каждое событие ассоциируется с динамическим объектом. В СТС входят все события, запланированные на текущий момент модельного времени. Прог-а управления моделированием в 1-ую очередь смотрит этот список и старается переместить по модели те динамич-е объекты, д\кот-ых выполнены условия. Если в списке таких объектов нет, процесс управления моделированием обращается к СБС. Он переносит все события, кот-е запланированы на ближайший мом-т модельного времени, из этого списка в СТС и повторяет его просмотр. При моделировании модель­ное время может меняться быстрее или медленнее, чем в реальной сис­т-е. Это зависит от степени детализации модели и сложности описа­ния изучаемого процесса. Модельное время изменяется при выполнении некот-ых событий, а события в сист-е моделирова­ния возникают в результате перемещения динамических объектов. Причиной изме­нения модельного времени может послужить явная задержка динамического объекта на некот-ый отрезок модельного времени.

Упрощенная схема протяжки модельного времени.

1- Начало моделирования;

2- СТС пуст?

3- Продвижение активного объекта в модели;

4- Движение активного объекта прекращено;

5- Извлечение динамического объекта из СБС с ближайшим временем активизации;

6- Изменение текущего модельного времени;

7- Поместить все объекты со временем выхода, равным тек. модельному времени, из СБС в СТС;

8- Конец моделирования?

9- Нормальное завершение моделирования.

2

3

1

9

завершение моделирования

6

7

4

5

8

моделиро-вания?

1. **Объектно-ориентированная моделирующая система.**

В большинстве случаев с помощью имитационных моделей исследуются характеристики и поведение системы на определенном отрезке времени, поэтому для моделирующей системы необходимым является реализация **двух функций**:

1. предоставление средств для формализованного описания системных компонентов, дисциплин выполнения различных работ, для задания структуры модели, привязки объектов модели к временной и пространственной координате;
2. осуществление координации событий, определение путей прохождения динамических объектов модели, изменение состояний узлов и передачу управления между блоками модели.

Существует несколько основных понятий, являющихся общими для большинства современных моделирующих систем:

1. **Граф модели.** Все процессы независимо от количества уровней структурного анализа, объединяются в виде направленного графа (многослойный иерархический).
2. **Динамические объекты.** Это объекты, моделирующие формальный запрос на какое-либо обслуживание.

Динамический объект может выполнять следующие действия:

* Порождать группы (семейства) других динамических объектов (GPSS, Split);
* Поглощать другие динамические объекты конкретного семейства;
* Захватывать ресурсы и использовать их некоторое время, а затем освобождать;
* Определять времена обслуживания, накапливать информацию о пройденном пути и иметь информацию о своем дальнейшем пути и о путях других динамических объектов.

Основные параметры динамических объектов:

* Уникальный идентификатор объекта;
* Идентификатор (номер) семейства, к которому принадлежит объект;
* Наборы различных ресурсов, которые динамический объект может захватывать и использовать какое-то время;
* Время жизни динамического объекта;
* Приоритет – неотрицательное число;
* Параметры обслуживания в каком-либо обслуживающем устройстве.

1. **Узлы графа модели** представляют собой центры обслуживания динамических объектов. В узлах динамические объекты могут задерживаться, обслуживаться, порождать семейства новых динамических объектов, уничтожать другие динамические объекты. Динамический объект всегда принадлежит одному из узлов графа модели.
2. **Событие** – факт входа или выходы из узла одного динамического объекта. Функция управления событиями отдана специально управляющей программе – координатору, автоматически внедряемому в состав модели.
3. **Ресурс** независимо от его природы в процессе моделирования может характеризоваться тремя общими параметрами: мощностью, остатком и дефицитом. Мощность – это макс. число ресурсных единиц. Остаток ресурса – число незанятых на данный момент единиц. Дефицит ресурса – количество единиц ресурса в суммарном запросе динамических объектов, стоящих в очереди к данному ресурсу.
4. **Пространство** – географическое, декартова плоскость. Узлы, динамические объекты и ресурсы могут быть привязаны к точкам пространства и мигрировать в нем.

Программная реализация моделирующих систем использует объектно-ориентированный способ представления процессов. Динамические объекты (транзакты), узлы, события и ресурсы являются основными объектами имитационной модели.

1. **Событийный и процессно-ориентированный подход к построению моделей.**

Событие – послед-сть логически связанных действий, происходящих в некот фиксированный момент мод-го времени. Появление события связ с изм-ем сост-я модельной среды.

**Событийный** подход осн на формировании потока событий. Такой поток образует сгруппиров. Последоват-сти действий:



Группы событий последоват-но, происх-щих в один момент вр, условно заключены в скобки.

Языки, реализующие **процессно-ориентир.подход**, имеют в своем составе блоки или операторы, позволяющие описать процесс продвижения компонентов ч\з с-му. В моделях, исп-щих подобную схему, описыв-ся не события и условия их возник-я, а процесс, развивающ-ся в ней.

1-создавать транзакты 20-25 ед мод времени; → 2-занести в 1-ый параметр тр-та знач-я; → 3-отправить на обраб-ку; → (А)

4-создавать транзакты 25-30 ед мод времени; → 5=2; → 6=3; → (А)

(А) = занять очередь → 7-занять прибор обраб-ки; → 8-задержаться на время обраб-ки; → 9-освободить прибор; → 10-уничтожить активный транзакт;

Оба подхода имеют как достоинства, так и недостатки. К достоинствам процессно-ориентир.представления моделей следует отнести компактность и наглядность.

Событийн. модели обладают большей гибкостью, но они уступают процессно-ориентиров.системам в простоте и наглядности составления моделей.

**Агентное моделирование-**метод, исследующ. поведение децентрализованных агентов и то, как такое поведение определяет поведение всей системы в целом. В отличие от системной динамики аналитик определяет поведение агентов на индивид. уровне, а глобальное поведение возникает как результат деят-сти множ-ва агентов (моделир-е «снизу вверх»).

Агентное моделир-е вкл. в себя элементы теории игр, сложных систем, мультиагентных систем и эволюционного программир-я, методы Монте-Карло, использует случ. числа

1. **Инструментальные средства моделирования**

Универс. Инструм.ср-вом создания моделей яв-ся языки программир. общего пользования (Pascal, C/C++ и др.). Кроме того, сущ-ет множ-во специализир.средств моделир-я, позволяющих быстрее и с меньшими затратами создавать и исследовать модели. В развитии специализир. Ср-в можно выделить **два направления**: **1.** Ср-ва моделир-я для анализа достаточно широкого класса систем: языки имитац.моделир-я (GPSS, SIMSCRIPT и др.), пакеты приклад. программ, использующих для моделир-я аналитич. методы, такие как MathCad, MathLab, SAS и др. Основной недостаток: их применение требует от исследователя спец.подготовки.**2.** Програм. комплексы, специализирующиеся на моделир-нии узкого круга систем одной конкретной предметной области. Недостаток: ограниченность применения таких программ одной предметной областью.Преимущ-ва: легкость их освоения специалистами в данной предмет.области, и эффективность применения вследствие узкой специализации.

В кажд. цикле создания программ. модели можно выделить **этапы**:**1. Формулирование проблемы**: описание исслед. проблемы, установление границ и ограничений моделируемой системы, определение целей исследования.**2. Разработка модели**: переход от реальной системы к некоторой логической схеме (абстрагирование).**3. Подготовка данных**: отбор данных, необх. для построения модели, и представление их в соответств. форме.**4. Трансляция модели**: описание модели на языке имитац. Моделир-я.**5. Оценка адекватности**: повышение до приемлемого уровня степени уверенности, с кот. можно судить относительно корректности выводов о реальной системе, получ. на основании обращения к модели.**6. Планирование**: определение условий проведения машинного эксперимента с имитац.моделью.**7. Экспериментирование**: многократн. прогон имит. модели на компьютере для получения требуемой инф-ии.**8. Анализ результатов**: изучение рез-тов для подготовки выводов и рекомендаций по решению проблемы.**9. Реализация и документирование**: реализация рекомендаций, получ.на основе имитации, и составление документации по модели и ее использованию.

1. **Система имитационного моделирования AnyLogic**

AnyLogic– уникальный инструмент имит.моделир-я, поддерживающ. на единой платформе абсолютно все существующ. подходы дискретно-событийного и непрерывного моделир-я. AnyLogicимеет развитый базовый язык дискретного и смешан. дискретно-непрерывного моделир-я.Библиотека AnyLogicEnterprise Library предоставляет высокоуровнев. интерфейс для быстрого создания дискретно-событийных моделей с пом. блок-схем. Графич. представление систем с пом. блок-схем широко исп-ся во многих важных сферах деятельности: произв-ве, логистике, системах обслуживания, бизнес-процессах, моделир-ии компьют. и телекоммуникацион. сетей.

Класс Entity яв-ся базовым классом для всех сообщений, кот. посылаются между активными объектами библиотеки Enterprise Library.

Под заявкой в библиотеке Enterprise Library может пониматься:

• заявка в ее обычном понимании (продукт, потребитель, пакет данных, документ),

• ресурс (оператор, машина, критическая секция),

• транспортер (поезд, автобус, корабль, автопогрузчик).

Заявки в их традиц. понимании генерир-ся объектами Source, проходят через смоделир. систему, где они обрабатыв-ся, обслужив-ся, транспортир-ся, конкурируют за право обладания ресурсами и эту систему покидают.

Ресурсы, созданные объектами Resource, м б.заняты заявками для выполнения каких-то задач, после чего они освобожд-ся и возвращ-ся в объект Resource.

Транспортеры, так же как и обычные заявки, создаются объектами Source, затем передаются в объекты Node и исп-ся для транспортировки других заявок между узлами и вдоль сегментов сети.

Объект класса Entity может исп-ся в любой из этих ролей, и при необходимости даже менять свою роль во время работы модели.

**Source** - Источник заявок. Обычно исп-ся в кач.начальной точки потока заявок, или как генератор ресурсов, транспортеров.

**Sink -** Уничтожает поступивш. заявки. Обычно исп-ся в кач.конечной точки потока заявок.

**Enter -** Пересылает заявки, переданные этому объекту либо “явно” через входной порт inputExternal, либо с помощью функции объекта take(), дальше по блок-схеме.

**SelectOutput -** Принимает заявку, и затем, в зависимости от заданного условия, передает ее на один из двух выходных портов.

**Queue -**  моделирует очередь, он хранит поступающ. заявки в опред.порядке: FIFO (заявки помещ-ся в очередь в порядке поступления), LIFO (заявки помещ-ся в порядке, обратном поступлению), RANDOM (заявки помещ-ся в произвольные места очереди) или PRIORITY (заявки помещ-ся в очередь в соответствии со значением своих полей priority).

1. **Общие понятия сетей Петри**

Сетевые модели (сети Петри) исп-ся для анализа причинно-следственных связей в сложных системах. Аппарат теории сетей Петри позволяет описывать стр-ру и взаимод-е параллельных систем и процессов. Сеть Петри (N-схема) задается 4мя элементами:N = <B,D,I,O>,где B - конечное множество позиций; D - конечное множество переходов; I - входная функция (прямая функция инцидентности), I:BXD→{0,1}; O - выходная функция (обратная функция инцидентности), O:DXB→{0,1}.

Iотображает переход dj в множество входных позицийbiϵI(dj), а выходная функция О отображает переход dj в множество выходных позиций biϵD(dj). Для каждого djϵDможно определить множество входных позиций перехода I(dj) и выходных позиций перехода O(dj) какI(dj)={biϵB|I(bi,dj)=1}, O(dj)={biϵB|I(dj,bi)=1}.

Аналогично, для каждого biϵB можно определить множество входных переходов позиции I(bi) и выходных переходов позицииO(bi):I(bi)={djϵD|I(dj,bi)=1}, O(bi)={djϵD|O(bi,dj)=1}.

Графически сеть Петри изображается в виде двудольного ориентированного мультиграфа, представляющего собой совокупность позиций и переходов.

Для представления динамич. Св-в объекта вводится ф-я маркировки (разметки) M:B🡪{0,1,2,…}. Маркировка - присвоение абстрактных объектов, называемых метками (фишками), позициям:NM = <B,D,I,O,M>.

Функционир-е сети Петри отражается путем перехода от разметки к разметке. M0 - начальная разметка. Смена разметок - срабатывание одного из переходов сети. Необходимое условие срабатывания перехода dj: biϵI(dj) {M(bi)≥1}, где M(bi) – разметка позиции bi.

Срабатывание перехода dj изменяет разметку сетиM(b) на M’(b) по следующему правилу:M’(b)=M(b)-I(dj)+O(dj), т.е. переход djизымает по одной метке из каждой своей входной позиции и добавляет по одной метке в каждую из выходных позиций.

Сети Петри предст. удобный математич. аппарат для моделир-я параллел.технологич. процессов с разделяемыми ресурсами. Преимущ-вом сетей Петри явл-ся легкость построения иерархич. конструкций, что позволяет сначала исследовать отдельные подсистемы, а затем всю систему в целом. Модели, построенные на основе сетей П., предназначены для анализа с пом. имитации на компьютере. Такие модели довольно легко реализуются программно даже с помощью универс. языков программир-я.